MULTILEVEL ACCELERATION OF NEUTRON TRANSPORT CALCULATIONS

A Thesis
Presented to
The Academic Faculty

by

José Ignacio Márquez Damián

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Nuclear Engineering in the
School of Mechanical Engineering

Georgia Institute of Technology
December 2007

MULTILEVEL ACCELERATION OF NEUTRON TRANSPORT CALCULATIONS

Approved by:

Weston M. Stacey, Committee Chair
Nuclear and Radiological Engineering
Georgia Institute of Technology

Cassiano R. E. de Oliveira, Advisor
Nuclear and Radiological Engineering
Georgia Institute of Technology

Nolan Hertel
Nuclear and Radiological Engineering
Georgia Institute of Technology

Wilfred Van Rooijen
Nuclear and Radiological Engineering
Georgia Institute of Technology

Date Approved: August 20, 2007

To Ana.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Cassiano de Oliveira, for his guidance and support during my stay at Georgia Tech. Working with him for the last year and a half was a very interesting experience, from both an academical and personal perspective. I would also like to extend my gratitude to Dr. HyeongKae Park for the many hours he spent explaining the theory that support this thesis, analyzing and discussing the methods, and debugging the code. He was virtually a co-advisor, and I would not probably have been finished without his help.

I also appreciate the help from Dr. Wilfred Van Rooijen during the last months, providing me access to the computational resources needed to develop and test the software, and solving many problems while I was working off campus.

I would also like to thank the fellow students at Nuclear and Radiological Engineering department for their friendship, and for making me fell a little bit less away from home.

This project was economically supported in part by Georgia Tech Research Institute and Oak Ridge National Laboratory.

And last, but not least, I would like to say thanks to Ana, whose infinite patience and unconditional support made me possible to be here.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

χ	Fission spectrum.
Γ	Boundary.
$\hat{\Omega}$	Direction unit vector.
$\hat{n}$	Normal unit vector.
$\mathbf{A}$	Coefficient matrix.
$\mathbf{e}$	Error vector.
$\mathbf{r}$	Residual vector.
$\mathbf{x}$	Solution vector.
$\mathfrak{I}_c^f$	Interpolation operator from mesh c to mesh f.
$\mathfrak{R}_f^c$	Restriction operator from mesh f to mesh c.
μ_0	Cosine of the scattering angle.
ν	Average number of neutrons emmited per fission.
ν_i	Number of relaxation iterations for step i.
ϕ	Scalar flux.
ψ	Angular flux.
ψ_g	Group flux.
σ	Macroscopic cross section.
θ_0	Scattering angle.
E	Energy.
h	Meshsize.
k, k_{eff}	Multiplication factor.
n	Angular density.
Q_g	Multigroup source.
R_i	Reaction rate for type i reactions.
S	External source.
P_N	Spherical harmonics expansion of the angular flux.
S_N	Discrete ordinates expansion of the angular flux.

SUMMARY

Nuclear reactor design requires the calculation of integral core parameters and power and radiation profiles. These physical parameters are obtained by the solution of the linear neutron transport equation over the geometry of the reactor. In order to represent the fine structure of the nuclear core a very small geometrical mesh size should be used, but the computational capacity available these days is still not enough to solve these transport problems in the time range (hours-days) that would make the method useful as a design tool. This problem is traditionally solved by the solution of simple, smaller problems in specific parts of the core and then use a procedure known as homogenization to create average material properties and solve the full problem with a wider mesh size. The iterative multi-level solution procedure is inspired in this multi-stage approach, solving the problem at fuel-pin (cell) level, fuel assembly and nodal levels. The nested geometrical structure of the finite element representation of a reactor can be used to create a set of restriction/prolongation operators to connect the solution in the different levels. The procedure is to iterate between the levels, solving for the error in the coarse level using as source the restricted residual of the solution in the finer level. This way, the complete problem is only solved in the coarsest level and in the other levels only a pair of restriction/interpolation operations and a relaxation is required.

In this work, a multigrid solver is developed for the in-moment equation of the spherical harmonics, finite element formulation of the second order transport equation. This solver is implemented as a subroutine in the code EVENT. Numerical tests are provided as a standalone diffusion solver and as part of a block Jacobi transport solver.

CHAPTER I

INTRODUCTION

1.1 Motivation

In radiation and nuclear reactor applications it is often necessary to compute integral and differential quantities, such as reaction rates or eigenvalues, that define the state of the system. One way to obtain these quantities is to derive and solve an equation for the intensity of the radiation field within the system. This method is known as *deterministic radiation transport*.

One of the difficulties of this approach is the dependence of the radiation field on six independent variables, each of which spans over several orders of magnitude. This problem is not particular to deterministic radiation transport and, as it appeared earlier in other fields of numerical modeling, many techniques have been developed to solve it. Since the late seventies the concept of multilevel methods has been developed, initially for simple boundary value problems, and now with applications in many fields specially fluid mechanics and material science.

This work analyzes the application of multilevel methods to the solution of radiation transport problems using EVENT, a P_N-FE solver of the second order, even parity transport equation.

1.2 Radiation transport and multilevel structure in nuclear reactor analysis

One of the most common applications of radiation transport is the calculation of nuclear reactors. Leaving aside the complexity of the transport equation itself, one of the biggest problems in its practical application is the presence of multiple scales in the different independent variables, namely: time, direction, position and energy:

- Time: from nuclear feedback effects ($\sim 10^{-7}$s) to fuel depletion ($\sim 10^{12}$s).

- Direction: from complete isotropy (fission emitted neutrons) to complete anisotropy

(beams).

- Position: from pin cell internals ($\sim 10^{-3}\mathrm{m}$), to full the extent of full reactor cores ($\sim 10^{1}\mathrm{m}$).

- Energy: from resonance widths ($\sim 10^{-3}\mathrm{eV}$) to neutron emission energies ($\sim 10^{7}\mathrm{eV}$)

To deal with this type of problems, averaging methods like cell homogenization and multigroup spectrum collapse have been developed through the years. Multilevel methods have been also applied to the problem, being angular multilevel [9] as *diffusion synthetic acceleration (DSA)* [3, 4] and *transport synthetic acceleration (TSA)* [12, 2] methods particularly popular to accelerate the converge of source iterations in the S_N methods. Spatial multigrid has been also applied to the 1D S_N [5] method and, to a lesser extent, to 2D and 3D S_N methods.

In the P_N approximation used in the EVENT code [6, 7] the terms of the angular expansion of the flux (represented as spherical harmonics) are orthogonal, rendering of no application the projection used in DSA. But even if we consider only time independent, one speed transport problems we still have to solve the challenge of spatial heterogeneity.

A typical reactor physics problem contains geometrical scales from the size of the nuclear radius ($\sim 10^{-14}\mathrm{m}$) to the external size of the core ($\sim 10^{1}\mathrm{m}$). Although the smaller scales (up to the material structure, $\sim 10^{-4}\mathrm{m}$) are usually taken into account in the cross section processing and therefore transparent to the reactor analyst, the remaining five orders of magnitude present in nuclear reactor problems make their solution a problem with a high computational cost.

Starting from the coarsest level we can identify the following hierarchy of levels:

- Full core level ($\sim 10^{1}\mathrm{m}$), with external boundary conditions that represent the effect of other materials and structures surrounding the core;

- fuel assembly level ($\sim 10^{-1}\mathrm{m}$), which represent the smallest unit in terms of fuel movement in, out and within the core;

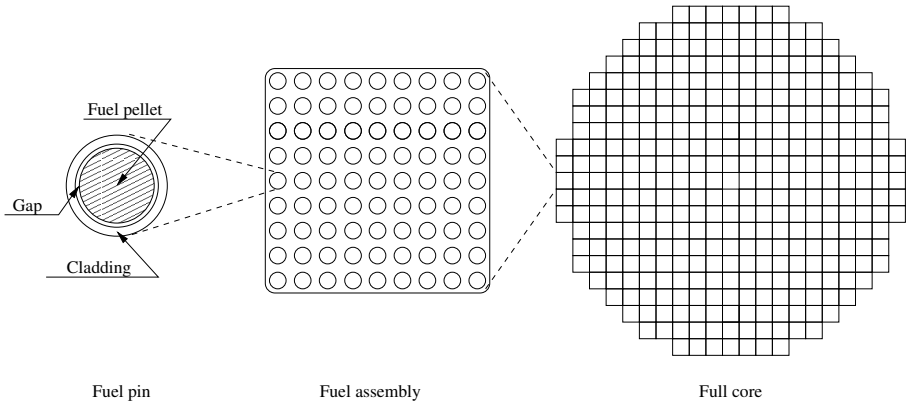


Figure 1: Hierarchical geometrical structure present in a nuclear power reactor.

- fuel pin level ($\sim 10^{-2}\mathrm{m}$), with the surrounding moderator and coolant represents the smallest repeated unit in terms of reactor analysis;

- fuel pin and absorbing components internals ($\sim 10^{-4} - 10^{-3}$) represent the smallest scale of interest in reactor analysis. They are still important for the spatial self-shielding effects that take place at this scale but at the same time the effect is usually too problem dependent to be introduced as a parameter in the cross section library.

The usual take on this problem is, as said before, to *homogenize* i.e. to average the properties on a region using as weight function the solution of an approximate problem solved with approximate boundary conditions. In multigrid, the averaged properties can be either obtained by homogenization or by application of the operators used to map one grid into the other. In any case, the transport equation is solved exactly only in the coarsest scale using average properties and a source given by the projected residual of the equation in a finer level. In all finer levels the only operations required are the application of a relaxation step to smooth the solution and the use of interpolation/relaxation operators to communicate the levels.

1.3 Objectives and thesis outline

The objective of this work is to implement a solver for the neutron transport equation, using the infrastructure of the code EVENT and a multilevel algorithm. That algorithm should use the structure of geometrical levels present in nuclear reactor problems to correct

the solution on the finest mesh.

To present this work, we first include some theory to support the numerical method chosen. Chapter 2 includes a brief analysis of the second order even neutron transport equation and its solution by the finite element, spherical harmonics method. Then, in Chapter 3, the general aspects of multigrid methods are introduced.

Chapter 4 analyzes aspects of the implementation of the diffusion an transport solvers, and results from numerical testing are shown and discussed in Chapter 5. Chapter 6 summarizes the conclusions and discusses some possible future work.

CHAPTER II

NEUTRON TRANSPORT AND THE CODE EVENT

2.1 First order neutron transport equation

2.1.1 General considerations

If we consider the neutrons to be point particles, and their energy to be enough to neglect quantum effects, their state can be represented by their position, $\vec{r}$, and velocity, $\vec{v}$. From an Eulerian point of view we can describe these neutrons by a distribution that represents the average density of neutrons in phase space, i.e.:

$$\begin{aligned} n(\vec{r},\hat{\Omega},E,t)\mathrm{d}\vec{r}\,\mathrm{d}\hat{\Omega}\,\mathrm{d}E = \quad & \text{Average number of neutrons at } \mathrm{d}\vec{r} \text{ about } \vec{r}, \\ & \text{moving in direction } \mathrm{d}\hat{\Omega} \text{ about } \hat{\Omega}, \\ & \text{with energy } \mathrm{d}E \text{ about } E, \text{ at time } t. \end{aligned} \tag{2.1}$$

where instead of the velocity, the direction $\hat{\Omega}$ and energy $E = 1/2mv^2$ of the particles were used. This distribution is known as *angular density*.

As the neutrons move through matter they interact with nuclei. These interactions can be characterized by a *reaction rate* R_i, which can be related to the angular density by introducing the concept of *cross section* σ [1] as the probability of interaction per unit path length traveled by the neutrons.

$$\begin{aligned} & R_i(\vec{r},\hat{\Omega},E)\mathrm{d}\vec{r}\,\mathrm{d}\hat{\Omega}\,\mathrm{d}E = \\ & \sigma_i(\vec{r},\hat{\Omega},E)v(E)n(\vec{r},\hat{\Omega},E)\,\mathrm{d}\vec{r}\,\mathrm{d}\hat{\Omega}\,\mathrm{d}E \quad \text{Number of interactions of type } i \text{ caused at } \mathrm{d}\vec{r} \text{ about } \vec{r}, \\ & \qquad\qquad \text{by neutrons moving in direction } \mathrm{d}\hat{\Omega} \text{ about } \hat{\Omega}, \\ & \qquad\qquad \text{with energy } \mathrm{d}E \text{ about } E, \text{ at time } t. \end{aligned} \tag{2.2}$$

The product of the angular density by the velocity is known as the *angular flux* and is often the quantity used to characterize the neutron distribution in radiation transport.

[1] We will use the symbol σ for macroscopic cross sections.

Using equation (2.2) to calculate the different reaction rates for the possible neutron interactions and calculating a balance over a differential phase space volume, we can arrive to the *first order neutron transport equation*:

$$\begin{aligned}\frac{\mathrm{d}n}{\mathrm{d}t} = \frac{1}{v}\frac{\partial\psi}{\partial t} + \hat{\Omega}\cdot\vec{\nabla}\psi = \\ -\sigma\psi + \int_0^\infty \mathrm{d}E' \int_{4\pi} \mathrm{d}\hat{\Omega}'\sigma_s(E',\hat{\Omega}'\to E,\hat{\Omega})\psi(\vec{r},E',\hat{\Omega}',t) \\ + \frac{\chi(E)}{4\pi}\int_0^\infty \mathrm{d}E'\nu(E')\sigma_f(E')\phi(E') + S(\vec{r},E,\hat{\Omega},t)\end{aligned} \tag{2.3}$$

where σ is the total cross section, σ_s is the scattering cross section and σ_f is the fission cross section. Each fission produces in average ν neutrons, which energy distribution is given by χ, the fission spectrum. As fission cross section and neutron emission is considered isotropic, the fission term was simplified by using the definition $\phi \equiv \int_{4\pi}\mathrm{d}\hat{\Omega}\,\psi$ called *scalar flux*. An external source of neutrons, S, was also considered.

Aside of the conditions stated at the beginning of this section, this equation is a valid model of neutron transport when the following conditions are satisfied:

1. particles may be considered points;
2. there are no external forces acting over the particle between collisions;
3. particle-particle interactions can be neglected (particle density much smaller than nuclear density);
4. collisions may be considered instantaneous;
5. material properties are isotropic;
6. material properties are independent of the radiation field (linear problem).
7. particles emitted by fission can be considered to be produced in the same point and time the fission occurred.

These conditions are usually satisfied in most shielding and reactor physics applications[2].

This equation is complete with a set of initial and boundary conditions. The initial condition is the angular flux value at each point of the phase space at time $t = t_0$, whereas the most common boundary conditions used in conjunction with this equation are:

- Vacuum boundary condition: $\psi(\vec{r}, \hat{\Omega}) = 0$ for $\hat{\Omega} \cdot \hat{n} < 0$ and $\vec{r} \in \Gamma$;

- Known incident flux: $\psi(\vec{r}, \hat{\Omega}) = \psi_B$ for $\hat{\Omega} \cdot \hat{n} < 0$ and $\vec{r} \in \Gamma$;

- Reflective surface: $\psi(\vec{r}, \hat{\Omega}) = \psi(\vec{r}, \hat{\Omega}')$ for $\hat{n} \cdot \hat{\Omega} = -\hat{n} \cdot \hat{\Omega}'$, $(\hat{\Omega} \times \hat{\Omega}') \cdot \hat{n} = 0$ and $\vec{r} \in \Gamma$;

- White boundary condition: $\psi(\vec{r}, \hat{\Omega}) = \int_{2\pi^+} \psi(\vec{r}, \hat{\Omega}')\, \mathrm{d}\hat{\Omega}'$ for $\hat{\Omega} \cdot \hat{n} < 0$ and $\vec{r} \in \Gamma$.

2.1.2 Multigroup scheme and the one-speed transport equation

From here on we will consider only the steady state transport equation with external sources:

$$\hat{\Omega} \cdot \vec{\nabla}\psi + \sigma\psi = \int_0^\infty \mathrm{d}E' \int_{4\pi} \mathrm{d}\hat{\Omega}' \sigma_s(E', \hat{\Omega}' \to E, \hat{\Omega}) \psi(\vec{r}, E', \hat{\Omega}') + S(\vec{r}, E, \hat{\Omega}) \tag{2.4}$$

and the homogeneous k-eigenvalue equation [3]:

$$\hat{\Omega} \cdot \vec{\nabla}\psi + \sigma\psi = \int_0^\infty \mathrm{d}E' \int_{4\pi} \mathrm{d}\hat{\Omega}' \sigma_s(E', \hat{\Omega}' \to E, \hat{\Omega}) \psi(\vec{r}, E', \hat{\Omega}') + \frac{1}{k} \frac{\chi(E)}{4\pi} \int_0^\infty \mathrm{d}E' \nu(E') \sigma_f(E') \phi(E') \tag{2.5}$$

To deal with the energy dependency of the angular flux and the cross sections present in these equations one of the most used methods is the *multigroup* method. If we take equation (2.4) and integrate it over an energy range ΔE_g (figure 2) we get:

$$\int_{\Delta E_g} \mathrm{d}E \hat{\Omega} \cdot \vec{\nabla}\psi + \int_{\Delta E_g} \sigma\psi = \int_{\Delta E_g} \int_0^\infty \mathrm{d}E' \int_{4\pi} \mathrm{d}\hat{\Omega}' \sigma_s(E', \hat{\Omega}' \to E, \hat{\Omega}) \psi(\vec{r}, E', \hat{\Omega}') + \int_{\Delta E_g} S(\vec{r}, E, \hat{\Omega}) \tag{2.6}$$

[2] When the last requisite is not satisfied, an additional set of equations for the neutron emitter fission products is necessary. But, as we will focus our study on steady state problems, this is not a concern.

[3] In multiplicative systems this eigenvalue, known as *multiplication factor*, is used to measure the reactivity of the system

Then, applying the following definitions:

$$\begin{aligned} \psi_g &\equiv \int_{\Delta E_g} \mathrm{d}E\, \psi(E) \quad \sigma_g \equiv \frac{\int_{\Delta E_g} \mathrm{d}E\, \sigma(E)\psi(E)}{\psi_g} \\ \sigma_s^{g\to g'} &\equiv \frac{\int_{\Delta E_g} \mathrm{d}E \int_{\Delta E'_g} \mathrm{d}E'\, \sigma_g(E'\to E)\psi(E')}{\psi_g} \qquad S_g \equiv \int_{\Delta E_g} S \end{aligned} \tag{2.7}$$

to equation (2.6) we obtain:

$$\hat{\Omega}\cdot\vec{\nabla}\psi_g + \sigma_g\psi_g = \sum_{g'=1}^{G}\int_{4\pi} \mathrm{d}\hat{\Omega}'\sigma_s^{g'\to g}(\hat{\Omega}'\to\hat{\Omega})\psi_{g'}(\vec{r},\hat{\Omega}') + S_g(\vec{r},\hat{\Omega}) \tag{2.8}$$

This formulation conserves the reaction rate, as the product of the multigroup flux and cross sections is equal by definition to the reaction rate (equation (2.6)). But, in order to this formulation to be exact, the weight function used in the group averages must be the angular flux which at this point is till unknown. Instead a simplified, smaller problem with boundary conditions that represent the in an approximate way the complete system. This simplified problem can be either solved with a higher number of groups or a continuous energy model.

The scattering contribution (first term on the right hand side of equation (2.8)) can be further expanded to separate the contribution from the scattering within group g, and from other groups:

$$\begin{aligned} \hat{\Omega}\cdot\vec{\nabla}\psi_g + \sigma_g\psi_g &= \int_{4\pi} \mathrm{d}\hat{\Omega}'\sigma_s^{g\to g}(\hat{\Omega}'\to\hat{\Omega})\psi_g(\vec{r},\hat{\Omega}') \\ &+ \sum_{g'\neq g}\int_{4\pi} \mathrm{d}\hat{\Omega}'\sigma_s^{g'\to g}(\hat{\Omega}'\to\hat{\Omega})\psi_{g'}(\vec{r},\hat{\Omega}') + S_g(\vec{r},\hat{\Omega}) \end{aligned} \tag{2.9}$$

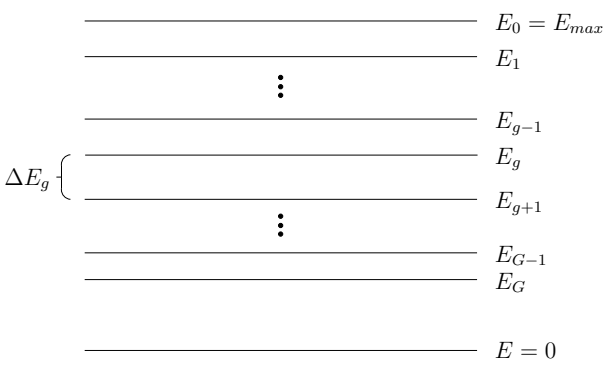


Figure 2: Group structure.

In a multigroup iterative scheme, the last two terms of equation (2.9) are considered a source Q_g with contributions from the actual external source and from in-scattering:

$$\hat{\Omega}\cdot\vec{\nabla}\psi_g + \sigma_g\psi_g = \int_{4\pi} \mathrm{d}\hat{\Omega}'\sigma_s^{g\to g}(\hat{\Omega}'\to\hat{\Omega})\psi_g(\vec{r},\hat{\Omega}') + Q_g \tag{2.10}$$

$$Q_g = \sum_{g'\neq g}\int_{4\pi} \mathrm{d}\hat{\Omega}'\sigma_s^{g'\to g}(\hat{\Omega}'\to\hat{\Omega})\psi_{g'}(\vec{r},\hat{\Omega}') + S_g(\vec{r},\hat{\Omega}) \tag{2.11}$$

In a similar way, if we introduce the definitions:

$$\begin{aligned} \psi_g &\equiv \int_{\Delta E_g}\mathrm{d}E\,\psi(E) & \phi_g &\equiv \int_{\Delta E_g}\mathrm{d}E\,\phi(E) \\ \sigma_g &\equiv \frac{\int_{\Delta E_g}\mathrm{d}E\,\sigma(E)\psi(E)}{\psi_g} & \sigma_s^{g\to g'} &\equiv \frac{\int_{\Delta E_g}\mathrm{d}E\int_{\Delta E'_g}\mathrm{d}E'\,\sigma_g(E'\to E)\psi(E')}{\psi_g} \\ (\nu\sigma_f)^g &\equiv \frac{\int_{\Delta E_g}\mathrm{d}E\,\nu(E)\sigma_f(E)\phi(E)}{\phi_g} & \chi_g &\equiv \int_{\Delta E_g}\mathrm{d}E\,\chi(E) \end{aligned} \tag{2.12}$$

to equation (2.5) and separate the scattering in its two components, we get:

$$\hat{\Omega}\cdot\vec{\nabla}\psi_g + \sigma_g\psi_g = \int_{4\pi} \mathrm{d}\hat{\Omega}'\sigma_s^{g\to g}(\hat{\Omega}'\to\hat{\Omega})\psi_g(\vec{r},\hat{\Omega}') + Q_g \tag{2.13}$$

$$Q_g = \sum_{g'\neq g}\int_{4\pi} \mathrm{d}\hat{\Omega}'\sigma_s^{g'\to g}(\hat{\Omega}'\to\hat{\Omega})\psi_{g'}(\vec{r},\hat{\Omega}') + \frac{1}{k_{eff}}\frac{\chi_g}{4\pi}\sum_{g'=1}^{G}(\nu\sigma_f)^{g'}\phi_{g'} \tag{2.14}$$

2.1.3 Angular expansion of the cross sections

Equations (2.10) and (2.13) are the *one speed transport equation*. For simplicity we will drop the g subscript, as all the quantities are referred to the same energy group:

$$\hat{\Omega}\cdot\vec{\nabla}\psi_g + \sigma\psi = \int_{4\pi} \mathrm{d}\hat{\Omega}'\sigma_s(\hat{\Omega}'\to\hat{\Omega})\psi(\vec{r},\hat{\Omega}') + Q \tag{2.15}$$

One of the assumptions made during the derivation of the transport equation is the material properties are isotropic, i.e. the cross section do not have an explicit dependence on $\hat{\Omega}$. In the case of the scattering cross section, this condition imposes the cross section is neither explicitly dependent on $\hat{\Omega}$ or $\hat{\Omega}'$, but on its scalar product, $\mu_0 = \hat{\Omega}\cdot\hat{\Omega}' = \cos\theta_0$ (figure 3).

Neutron scattering cross sections have a relatively small level of anisotropy. For this

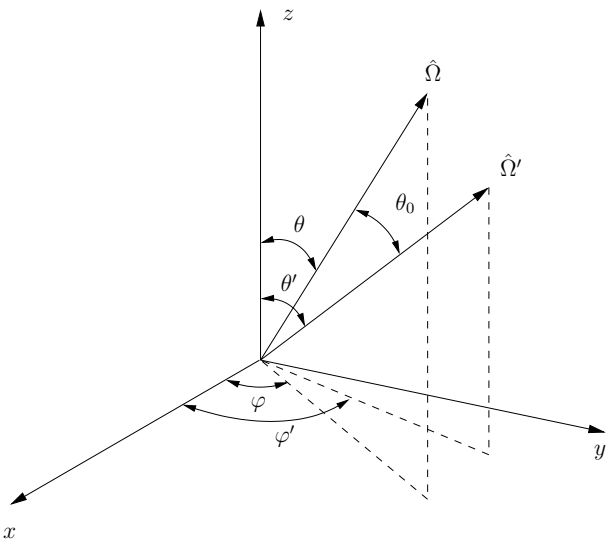


Figure 3: Definition of the scattering angle, θ_0.

reason scattering cross sections are usually expressed using a Legendre polynomial expansion on $\mu_0 = \cos\theta_0$:

$$\begin{aligned}\sigma_s(\vec{r}, \hat{\Omega} \to \hat{\Omega}') &= \sigma_s(\vec{r}, \hat{\Omega} \cdot \hat{\Omega}') = \sum_{l=0}^{\infty} \frac{2l+1}{4\pi} \sigma_{sl}(\vec{r}) P_l(\mu_0) \\ \sigma_{sl}(\vec{r}) &= 2\pi \int_{-1}^{1} \mathrm{d}\mu_0 \sigma(\mu_0) P_l(\mu_0)\end{aligned} \tag{2.16}$$

To combine the absorption and scattering terms we can use the following properties of the delta function:

$$\begin{aligned}\int_{4\pi} \mathrm{d}\hat{\Omega}'\, \delta(\hat{\Omega} - \hat{\Omega}') f(\hat{\Omega}') &= f(\hat{\Omega}) \\ \delta(\hat{\Omega} - \hat{\Omega}') &= \sum_{l=0}^{\infty} \frac{2l+1}{4\pi} (\vec{r}) P_l(\mu_0)\end{aligned} \tag{2.17}$$

to write the absorption term:

$$\begin{aligned}\sigma(\vec{r})\psi(\vec{r}, \hat{\Omega}) = \int_{4\pi} \mathrm{d}\hat{\Omega}'\, \delta(\hat{\Omega} - \hat{\Omega}') \sigma(\vec{r}) \psi(\vec{r}, \hat{\Omega}') = \\ \int_{4\pi} \mathrm{d}\hat{\Omega}' \sum_{l=0}^{\infty} \frac{2l+1}{4\pi} (\vec{r}) P_l(\mu_0) \sigma(\vec{r}) \psi(\vec{r}, \hat{\Omega}')\end{aligned} \tag{2.18}$$

Inserting this expansion back in equation (2.15) we get:

$$\hat{\Omega} \cdot \vec{\nabla} \psi(\hat{\Omega}) + \sum_{l=0}^{\infty} \frac{2l+1}{4\pi} \sigma_l \int_{4\pi} \mathrm{d}\hat{\Omega}' P_l(\mu_0) \psi(\vec{r}, \hat{\Omega}') = Q(\vec{r}, \hat{\Omega}) \tag{2.19}$$

with $\sigma_l = \sigma - \sigma_{sl}$.

In order to solve it numerically we still have to discretize space and angle, but this is going to be done in its even parity, second order form.

2.2 *Even parity transport equation*

Equation (2.19) is valid for all $\hat{\Omega}$, and particularly for $-\hat{\Omega}$,

$$-\hat{\Omega}\cdot\vec{\nabla}\psi(-\hat{\Omega})+\sum_{l=0}^{\infty}\frac{2l+1}{4\pi}\sigma_l\int_{4\pi}\mathrm{d}\hat{\Omega}'P_l(-\mu_0)\psi(\vec{r},\hat{\Omega}')=Q(\vec{r},-\hat{\Omega}) \tag{2.20}$$

Adding and subtracting equations (2.19) and (2.20) we obtain:

$$\hat{\Omega}\cdot\vec{\nabla}\psi^{-}(\hat{\Omega})+\sum_{l\,even}\frac{2l+1}{4\pi}\sigma_l\int_{4\pi}\mathrm{d}\hat{\Omega}'P_l(-\mu_0)\psi^{+}(\vec{r},\hat{\Omega}')=Q^{-}(\vec{r},\hat{\Omega}) \tag{2.21}$$

and

$$\hat{\Omega}\cdot\vec{\nabla}\psi^{+}(\hat{\Omega})+\sum_{l\,odd}\frac{2l+1}{4\pi}\sigma_l\int_{4\pi}\mathrm{d}\hat{\Omega}'P_l(-\mu_0)\psi^{-}(\vec{r},\hat{\Omega}')=Q^{+}(\vec{r},\hat{\Omega}) \tag{2.22}$$

where the even and parity components of the angular flux, $\psi^{\pm}$, and the source, $Q^{\pm}$, are given by:

$$\psi^{\pm}=\frac{1}{2}[\psi(\hat{\Omega})\pm\psi(-\hat{\Omega})] \tag{2.23}$$

$$Q^{\pm}=\frac{1}{2}[Q(\hat{\Omega})\pm Q(-\hat{\Omega})] \tag{2.24}$$

Let us now define the operators:

$$G^{-1}f(\hat{\Omega})=\sum_{l\,odd}\frac{2l+1}{4\pi}\sigma_l\int_{4\pi}\mathrm{d}\hat{\Omega}'P_l(\mu_0)f(\hat{\Omega}') \tag{2.25}$$

$$Cf(\hat{\Omega})=\sum_{l\,even}\frac{2l+1}{4\pi}\sigma_l\int_{4\pi}\mathrm{d}\hat{\Omega}'P_l(\mu_0)f(\hat{\Omega}') \tag{2.26}$$

and use them to express equations (2.21) and (2.22):

$$\hat{\Omega}\cdot\vec{\nabla}\psi^{+}(\hat{\Omega})+G^{-1}\psi^{-}(\vec{r},\hat{\Omega})=Q^{-}(\vec{r},\hat{\Omega}) \tag{2.27}$$

$$\hat{\Omega}\cdot\vec{\nabla}\psi^{-}(\hat{\Omega})+C\psi^{+}(\vec{r},\hat{\Omega})=Q^{+}(\vec{r},\hat{\Omega}) \tag{2.28}$$

As suggested by the notation, the inverse of G^{-1} exists [1]. Using this operator we can isolate the odd parity flux from equation (2.27):

$$\psi^{-}(\hat{\Omega})=G[Q^{-}(\hat{\Omega})-\hat{\Omega}\cdot\vec{\nabla}\psi^{+}(\hat{\Omega})] \tag{2.29}$$

and replace it in equation (2.28) to obtain the *even parity transport equation*:

$$-\hat{\Omega}\cdot\vec{\nabla}G\,\hat{\Omega}\cdot\nabla\psi^{+}+C\psi^{+}=Q^{+}(\vec{r},\hat{\Omega})-\hat{\Omega}\cdot\nabla GQ^{-} \tag{2.30}$$

The boundary conditions for this equation can be derived from the conditions applied and the properties of the even and odd parity fluxes. For instance, the vacuum boundary condition can be rewritten using equation (2.23) as:

$$\begin{gathered}\psi(\vec{r},\hat{\Omega})=\psi^{+}(\vec{r},\hat{\Omega})+\psi^{-}(\vec{r},\hat{\Omega})=0\\ \text{for } \hat{\Omega}\cdot\hat{n}<0 \text{ and } \vec{r}\in\Gamma\end{gathered} \tag{2.31}$$

2.3 Variational formulation of the even parity equation

The computational code EVENT uses a finite element spherical harmonics approximation of the even parity transport equation. This solution is obtained by the Ritz procedure, finding the solution by minimizing a functional.

This functional is given by:

$$\begin{aligned}K^{+}[\psi^{+}]=&\left(\hat{\Omega}\cdot\nabla\psi^{+},G\hat{\Omega}\cdot\nabla\psi^{+}\right)+\left(\psi^{+},C\psi^{+}\right)+\left\langle\psi^{+},\psi^{+}\right\rangle\\&-2\left(\psi^{+},Q^{+}\right)-2\left\langle\psi^{+},T\right\rangle-2\left(\hat{\Omega}\cdot\nabla\psi^{+},GQ^{-}\right)\end{aligned} \tag{2.32}$$

where $(.,.)$ denotes angular and volume integration over all angles and over all the domain:

$$(f,g)\equiv\int_{4\pi}\mathrm{d}\hat{\Omega}\int_{V}\mathrm{d}V\,fg \tag{2.33}$$

and $\langle .,.\rangle$ angular and surface integral over all angles and over the external surface:

$$\langle f,g\rangle\equiv\int_{4\pi}\mathrm{d}\hat{\Omega}\int_{\delta V}\mathrm{d}\Gamma\,|\hat{\Omega}\cdot\hat{n}|\,fg \tag{2.34}$$

Now, it can be shown [1, 7, 11] the condition to vanish its first variation is:

$$\begin{aligned}-\hat{\Omega}\cdot\vec{\nabla}G\,\hat{\Omega}\cdot\nabla\psi^{+}+C\psi^{+}&=Q^{+}(\vec{r},\hat{\Omega})-\hat{\Omega}\cdot\nabla GQ^{-} && \text{in the domain}\\ \psi^{+}+G[Q^{-}-\hat{\Omega}\cdot\nabla\psi^{+}]&=T && \text{on the external surface}\end{aligned} \tag{2.35}$$

which is the even parity transport equation with a prescribed flux T as boundary condition. The second variation of equation (2.32) is positive, making the minimizing function the best possible solution in a least-square sense.

2.4 *Spherical harmonics, finite element approximation*

In the Ritz procedure the minimization of equation (2.32) is performed by solving a linear system of linear equations. This system of equations is obtained approximating the even parity flux by a truncated series of trial functions:

$$\psi^{+}(\vec{r},\hat{\Omega}) \approx f(\vec{r},\hat{\Omega}) = \sum_{i} f_i u_i(\vec{r},\hat{\Omega}) = \mathbf{u}^T\mathbf{f} \tag{2.36}$$

and replacing it on the functional, which now can be written

$$K[f] = \mathbf{f}^T\mathbf{A}\mathbf{f} - 2\mathbf{f}^T\mathbf{s} \tag{2.37}$$

with

$$\mathbf{A} \equiv F[\mathbf{u},\mathbf{u}^T] \tag{2.38}$$

and

$$\mathbf{s} \equiv F_s[\mathbf{u}] \tag{2.39}$$

where the functionals F and F_s are given by:

$$\begin{aligned} F[f,g] &= \left(\hat{\Omega}\cdot\nabla f, G\hat{\Omega}\cdot\nabla g\right) + (f,Cg) + \langle f,g\rangle \\ F_s[f] &= \left(f,Q^{+}\right) + \langle f,T\rangle - \left(\hat{\Omega}\cdot\nabla f, GQ^{-}\right) \end{aligned} \tag{2.40}$$

Then, the requirement $\delta K[f] = 0$ is equivalent to the solution of the system:

$$\mathbf{A}\mathbf{f} = \mathbf{s} \tag{2.41}$$

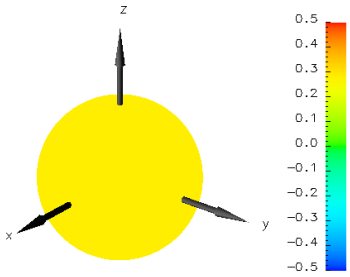
The basis functions $u_i(\vec{r},\hat{\Omega})$ are factorized into an angular component, represented by spherical harmonics, and a spatial component, represented with Lagrange finite elements:

$$u_i(\vec{r},\hat{\Omega}) = \sum_{e=1}^{E}\sum_{j=1}^{N_e}\sum_{m=1}^{M} q_m(\hat{\Omega})B_j^e(\vec{r}) = \sum_{e=1}^{E}\sum_{j=1}^{N_e}\sum_{l=0}^{N}\sum_{m=0}^{l} Y_{lm}(\hat{\Omega})B_j^e(\vec{r}) \tag{2.42}$$

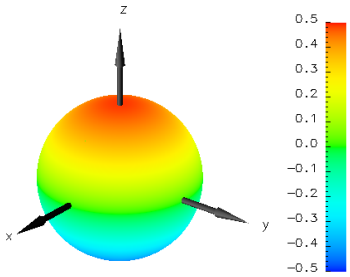
where the spherical harmonics with $m < 0$ are already considered by the symmetry of the even parity formulation.

$Y_{lm}(\hat{\Omega})$ are normalized spherical harmonics (figure 4), written in terms of trigonometric functions (instead of complex exponentials) on the azimuthal angle:

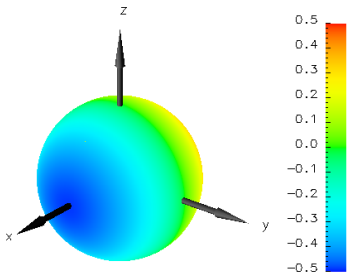
$$\begin{bmatrix} Y_l^o(\hat{\Omega}) \\ Y_l^e(\hat{\Omega}) \end{bmatrix} = \sqrt{\frac{2l+1}{4\pi}(2-\delta_{l,0})\frac{(l-m)!}{(l+m)!}}P_l^m(\mu) \begin{bmatrix} \sin m\varphi \\ \cos m\varphi \end{bmatrix} \tag{2.43}$$



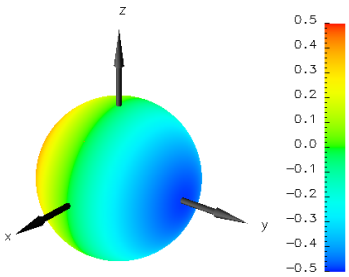
$$Y^e_{0,0} = \frac{1}{2}\sqrt{\frac{1}{\pi}}$$



$$Y^e_{1,0} = \frac{1}{2}\sqrt{\frac{3}{\pi}}\cos\theta$$



$$Y^e_{1,1} = -\frac{1}{2}\sqrt{\frac{3}{\pi}}\sin\theta\cos\varphi$$



$$Y^o_{1,1} = -\frac{1}{2}\sqrt{\frac{3}{\pi}}\sin\theta\sin\varphi$$

Figure 4: Spherical harmonics for $l = 0$ and $l = 1$.

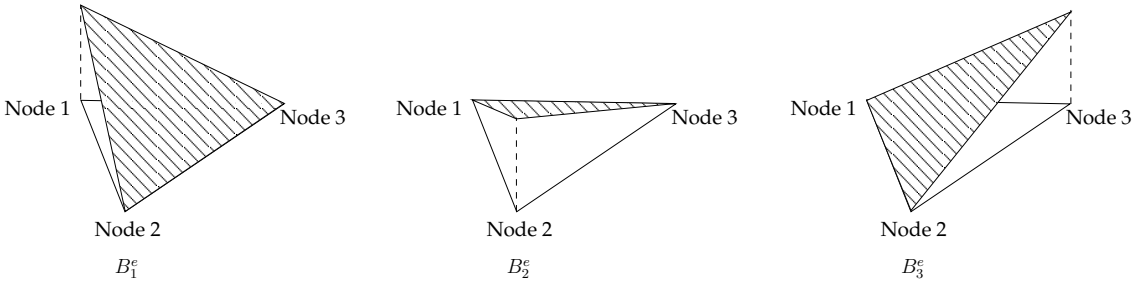


Figure 5: Linear triangular finite elements in two dimensions.

B_j^e are the Lagrange element shape functions. These functions are only non-zero inside of E non-overlapping regions (triangles or quadrilaterals in 2D) or *elements* and defined in such a way its value is 1 at one of the nodes of the element, and 0 on the rest. Figure 5 represents the three linear basis functions corresponding to the nodes of a triangular element in two dimensions.

2.5 Structure of the code EVENT

EVENT reads its data from a set of files produced by the preprocessor GEM [16]. These files are a run parameter file, a mesh file and a material data file which GEM produces acting as an interface from human readable data and several other data formats.

After reading the input file and preparing the storage arrays, EVENT calls the subroutine which assembles the coefficient matrix, prepares and constrains the source terms and solves the in-group system of equations (equation (2.41)) iterating over the groups if necessary. After the solution is converged, a postprocessor module calculates any requested integral quantities and produces problem-wide balances and checks.

Radiation transport problems are solved by a nested iterative method; from the outer to the inner loops:

- Multigroup loop (including eigenvalue and upscattering iterations);

- Moment iteration on the in-group equation;

- Spatial solver on the in-moment equation;

From a generic point of view this scheme of nested iterations for the solution of the multigroup even parity equation can be seen as the solution of a system of equations (Figure 6).

In this scheme, region (a) corresponds to the *downscattering*, which is solved by forward substitution as the submatrix is lower triangular. When the solver reaches region (b), iterations are required to solve the coupling caused by upscattering in the thermal region. If the problem includes fission, region (c) of the matrix will also be non-zero, requiring

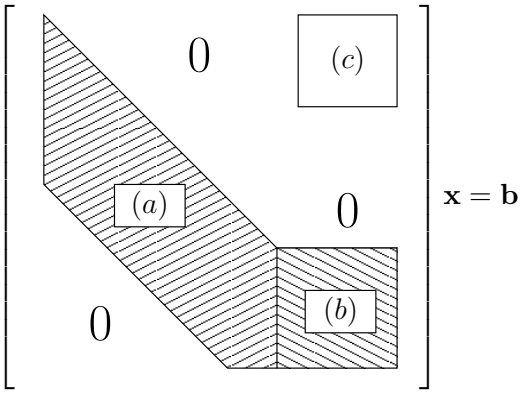


Figure 6: Scheme of a generic multigroup neutron transport problem.

iteration to solve the matrix, as the groups associated with fission spectrum are coupled with the thermal groups.

The matrix represented in Figure 6 is a block matrix, where each block represents the coefficient matrix associated to the discretized one-speed even parity transport equation (equation (2.41)). These matrices are symmetric, positive definite and have a dimension of $NM \times NM$, where N is the number of nodes and M is the number of angular moments.

These matrices are also composed of blocks: the A_{mm} diagonal blocks (figure 7) are $N \times N$ matrices assembled explicitly, whereas matrix vector multiplications involving the non-diagonal blocks are performed by the MATVEC subroutine which computes the coefficients on request, storing and reusing the spatial integrals.

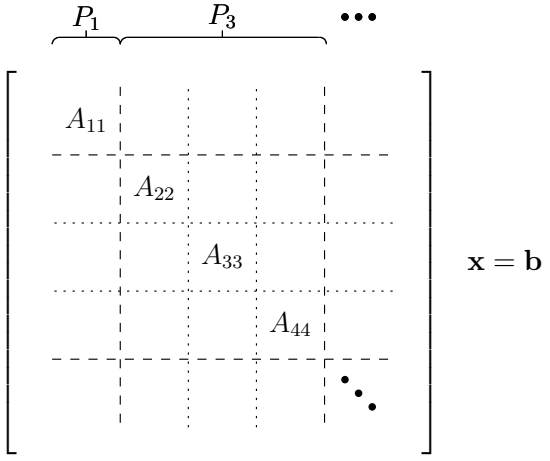


Figure 7: Scheme of the assembled in-group matrix of coefficients for a transport problem..

The standard solver of EVENT, the subroutine CGSOL5, uses the preconditioned conjugate gradients method [13]

CHAPTER III

MULTIGRID METHODS

3.1 Overview

In a system of linear equations the solution $\mathbf{x}$ satisfies:

$$\mathbf{A}\,\mathbf{x} = \mathbf{b} \tag{3.1}$$

When solving this system with an iterative method, iteration k yields an approximation of the solution $\mathbf{x}^{(k)}$ associated to an error $\mathbf{e}^{(k)}$:

$$\mathbf{e}^{(k)} \equiv \mathbf{x} - \mathbf{x}^{(k)} \tag{3.2}$$

If we replace this definition into equation (3.1) we obtain the *error equation*:

$$\mathbf{A}\,\mathbf{e}^{(k)} = \mathbf{r}^{(k)} \tag{3.3}$$

where $\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\,\mathbf{x}^{(k)}$ is the *residual* for the k-th iteration. Multigrid methods are based on the solution of this equation over a coarse grid, to find a linear correction to the solution on the fine grid. Doing this have two direct advantages:

- The computational cost of solving the equation on a coarser grid is smaller, and
- In a coarser grid iterative methods are more efficient, as the low frequency components of the error have a relative higher frequency on the coarse grid and are attenuated faster by the relaxation iterations.

We will go in more detail in these properties in the following sections.

In order to perform this linear correction, iterative methods require the implementation of three component parts:

- A relaxation sweep using a stationary iteration;
- A restriction/interpolation operator to communicate the different levels;

- A coarse coefficient matrix for each coarse grid; and

- An exact solver to find the error in the coarsest grid.

3.2 Relaxation iterations

If we split the coefficient matrix from equation (3.1) into its diagonal, upper, and lower parts:

$$\mathbf{A} = \mathbf{D} - \mathbf{L} - \mathbf{U} \tag{3.4}$$

the matrix equation (3.1) becomes

$$(\mathbf{D} - \mathbf{L} - \mathbf{U})\,\mathbf{x} = \mathbf{b} \tag{3.5}$$

From this we can isolate the diagonal elements of $\mathbf{A}$ and write:

$$\begin{aligned} \mathbf{D}\,\mathbf{x} &= \mathbf{b} + (\mathbf{L} + \mathbf{U})\,\mathbf{x} \\ \mathbf{x} &= \mathbf{D}^{-1}\,\mathbf{b} + \mathbf{D}^{-1}\,(\mathbf{L} + \mathbf{U})\,\mathbf{x} \end{aligned} \tag{3.6}$$

This equation is exact, but is still implicit. Now, evaluating the matrix-vector product in the right hand side using the value of $\mathbf{x}$ from a previous iteration we get the *Jacobi iteration*:

$$\mathbf{x}^{(k)} = \mathbf{D}^{-1}\,\mathbf{b} + \mathbf{D}^{-1}\,(\mathbf{L} + \mathbf{U})\,\mathbf{x}^{(k-1)} \tag{3.7}$$

Returning to equation (3.5) we can use the split coefficient matrix to find:

$$\begin{aligned} (\mathbf{D} - \mathbf{L})\mathbf{x} &= \mathbf{b} + \mathbf{U}\mathbf{x} \\ \mathbf{x} &= \mathbf{D}^{-1}(\mathbf{b} + \mathbf{U}\mathbf{x} + \mathbf{L}\mathbf{x}) \end{aligned} \tag{3.8}$$

This is formally equal to the equation obtained for the Jacobi iteration. But, as the matrix vector product $\mathbf{L}\mathbf{x}$ on the right hand side only requires the use of precalculated elements of $\mathbf{x}$ we can overwrite the vector to do:

$$x_i^k = \tfrac{1}{a_{ii}} \left(b_i + \textstyle\sum_{j=i+1}^{N} a_{ij} x_j^{k-1} + \sum_{j=1}^{i-1} a_{ij} x_j^k \right) \tag{3.9}$$

This iteration is known as *forward Gauss-Seidel* as $\mathbf{x}$ is swept from i to N. If we reverse the order on which $\mathbf{x}$ is updated, the iteration obtained is *backward Gauss-Seidel*.

Both Jacobi and Gauss-Seidel constitute *stationary linear iterations*, as the rule used to update the solution does not change from one iteration to the next.

3.2.1 Spectral properties

The stationary iterations shown above can be written in a general way as:

$$\mathbf{x}^k = \mathbf{R}\mathbf{x}^{k-1} + \mathbf{g} \tag{3.10}$$

For Jacobi iterations $\mathbf{R} = \mathbf{D}^{-1}$, $\mathbf{D}^{-1}\mathbf{b}$, and for forward Gauss-Seidel $\mathbf{R} = (\mathbf{D} - \mathbf{L})^{-1}$, $\mathbf{g} = (\mathbf{D} - \mathbf{L})^{-1}\mathbf{b}$. Noting that for the true solution:

$$\mathbf{x} = \mathbf{R}\mathbf{x} + \mathbf{g} \tag{3.11}$$

we can substract equation (3.10) from (3.11) to obtain:

$$\mathbf{e}^k = \mathbf{R}\mathbf{e}^{k-1} \tag{3.12}$$

Then, the error for the k-th iteration is equal to k applications of the operator $\mathbf{R}$ to the error of the guess, $\mathbf{e}^{(0)}$. If we use the eigenvectors of $\mathbf{R}$ to expand $\mathbf{e}$ we can write:

$$\mathbf{e}^k = \mathbf{R}^k\mathbf{e}^{k-1} = \sum_{m=1}^{n-1} \mathbf{R}^k c_m \mathbf{w}_m = \sum_{m=1}^{n-1} \lambda_m^k(\mathbf{R}) c_m \mathbf{w}_m \tag{3.13}$$

Then, the error is not attenuated uniformly. Each component, expanded using the eigenvectors of $\mathbf{R}$ as base, is attenuated by a factor given by the corresponding eigenvalue. The maximum value of λ_m is the *spectral radius* for $\mathbf{R}$ and determines the asymptotic convergence rate of the method.

It can be shown [15] the higher eigenvalues of $\mathbf{R}$ are associated to the lower frequency eigenvectors. Then, the successive application of $\mathbf{R}$ eliminates the high frequency components of the error fast, leaving the lower frequency modes.

This property can be easily shown with a numerical experiment. Using a homogeneous diffusion problem ($\sigma = 0.50$, $\sigma_a = 0.05$, $\sigma_s = 0.45$) over a 128×128 mesh using Jacobi iterations and a random guess, we can see 8 the convergence stalls after the first 10 iterations. Plotting the error after 1 (figure 9, 10 (figure 10 and 100 (figure 10 iterations we can see how the high frequency modes of the error are eliminated in the first iterations, but many more are needed to deal with the remaining low frequency modes.

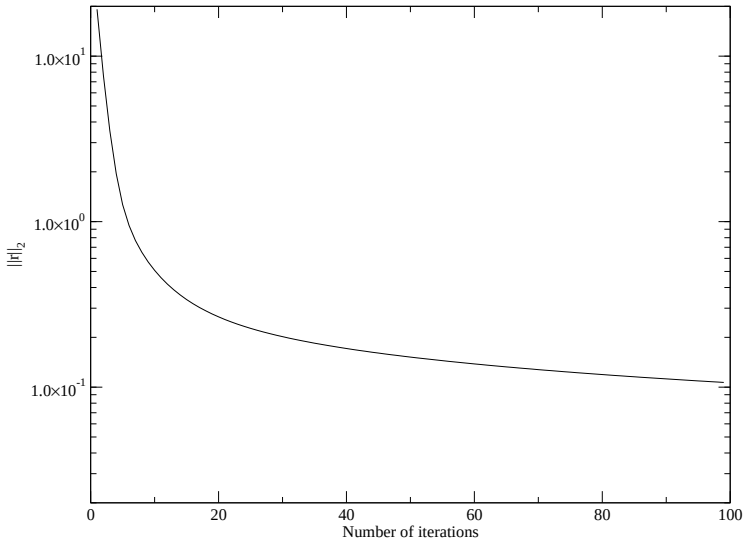


Figure 8: Convergence of a homogeneous diffusion problem ($\sigma = 0.50$, $\sigma_a = 0.05$, $\sigma_s = 0.45$) over a 128×128 mesh using Jacobi iterations and a random guess.

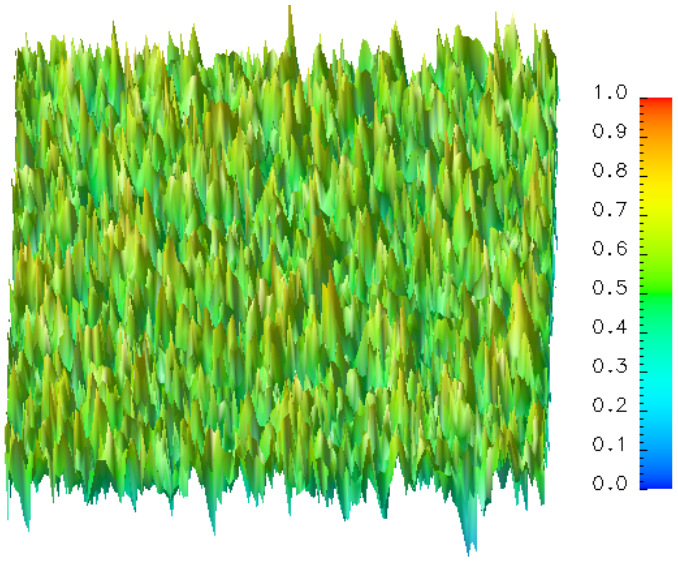


Figure 9: Error after 1 iteration.

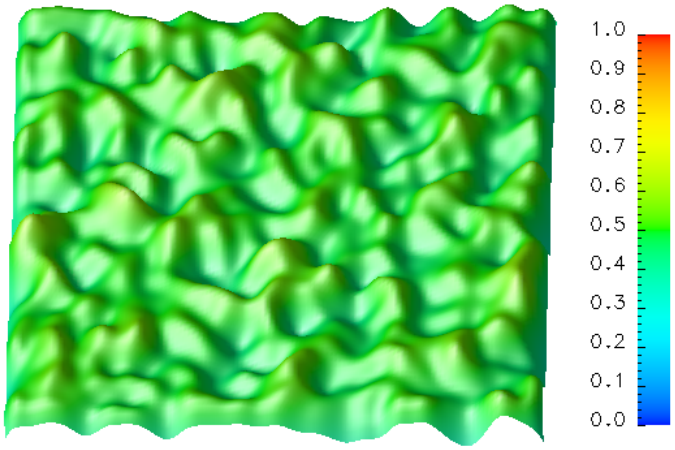


Figure 10: Error after 10 iterations.

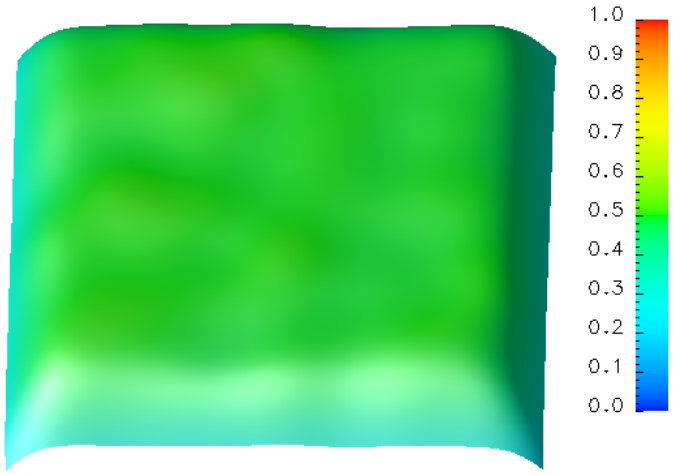


Figure 11: Error after 100 iterations.

3.3 Restriction / interpolation operators

After a few relaxation iterations on the mesh with gridsize h_0, high frequency modes have been damped and convergence slows down. At this point the problem should be transfered into a coarser grid ($h_1 > h_0$), to solve equation (3.3). In order to do that the residual calculated at h_0 must be projected or *restricted* into h_1.

The simplest way to perform this task is by *injection*, i.e. by assigning to a point in the coarse grid the value in the corresponding point of the fine grid (figure 12).

Albeit this method is simple, it does not transfer any information from the neighbouring points. To achieve that we can assign to the coarse point a weighted average of the corresponding surrounding points in the fine grid. Under this concept two different operators are usually used: *full weighted* (figure 13), that averages all the surrounding points, and *half weighted* (figure 14), that only averages the points that lay on the directions determined by the coordinate system.

As it can be seen in figure 15, the projection to the coarse mesh of a rather smooth function in the fine mesh makes the function look more oscillatory. The explanation to that is the restriction operation preserves the wave number of the basis eigenfunctions that can be used to represent the vector. As the maximum wavenumber that can be represented on a given grid is limited by the number of nodes, an eigenfunction with the same wavenumber is relatively more oscillatory (i.e. it takes a higher possible wavenumber) in a coarser grid.

This, combined with the fact that the more oscillatory mode in the coarser grid is dampen more efficiently by a relaxation iteration, makes the combination restriction/relaxation an optimal way to attenuate high and low wavenumber modes. Moreover, if we try to restrict a very oscillatory mode (a mode with wavenumber higher than the number of nodes of the coarse grid) we will see (figure 16) it is misrepresented as a smooth mode. This phenomenon is known as *aliasing*.Then, the smoothing step previous to the restriction is not only a way to take advantage of the higher convergence rate of the relaxation iterations, but also a way to eliminate modes could be hidden by the restriction operator.

As we defined an operator that projects a vector represented on a fine grid into a coarse

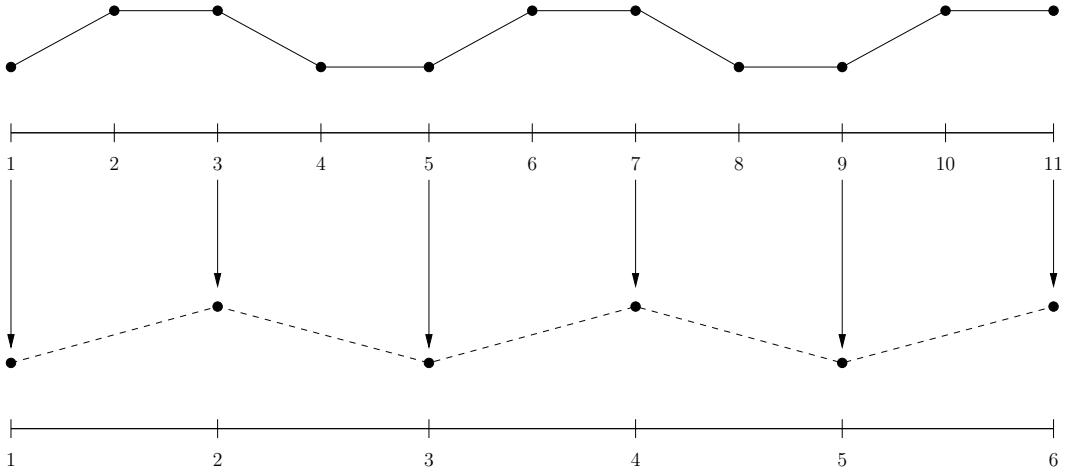


Figure 12: Restriction by injection of a fine-grid vector into a coarse grid.

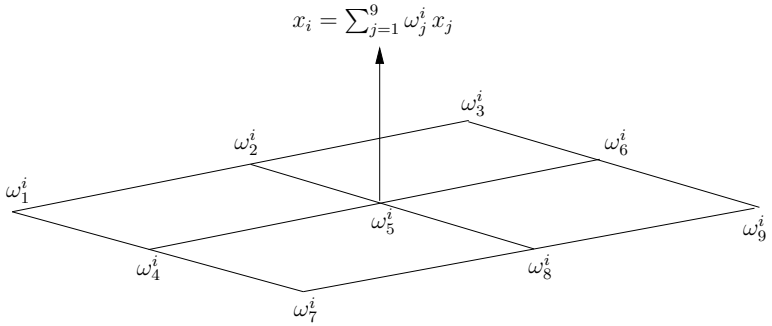


Figure 13: Restriction by full weighting.

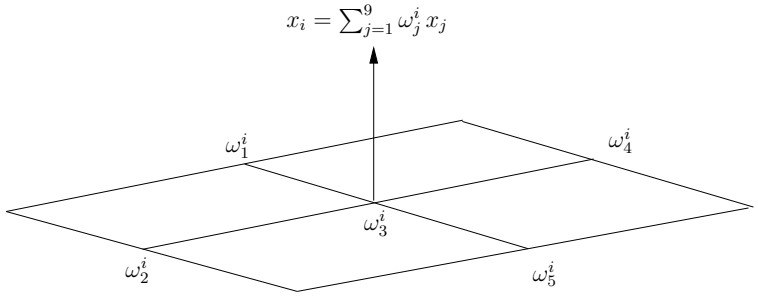


Figure 14: Restriction by half weighting.

grid, we also need to define an operator that transfers the calculated error back into the fine grid to use it as a correction. Such operation is an *interpolation* or *prolongation*, and can be performed using any interpolation methods (linear, quadratic, splines). As we will see, linear interpolation is usually enough.

After we performed a smoothing step over the fine grid, the only modes remaining of the error are smooth because the inefficiency of the relaxation iteration on low frequency modes. The solution of the error equation on the coarse grid would yield a function that is smooth in the fine grid, and then it can be satisfactorily represented on the coarse grid and easily interpolated. Being the function smooth, a low order interpolating polynomial is enough to perform the task.

A linear interpolation is the opposite operation to a full weighted restriction operator where the weights were found using linear functions. Then, the same operator can be used, transposed and scaled, to perform the interpolation.

3.4 Coarse grid coefficient matrix

After relaxing the solution in the finer grid and project the residual into the coarse grid we can either relax again and continue going deeper into coarser grids, or solve the error equation to find the additive correction for the fine level. Both operations require the definition of a coefficient matrix for each coarse level.

This coarse grid coefficient matrix can be defined either geometrically, by applying the restriction/interpolation operators on the fine grid matrix (a process known as variational coarsening) or using a physical interpretation of the coarse grid and assembling the matrix using homogenized properties.

3.5 V-cycle

At this point we have all the components to perform a multilevel solution of a system of equations. First, we apply one or more relaxation iterations to smooth out the high frequency components of the error. Then, we compute the residual and project it into a coarse grid using the restriction operator. If we defined more than two levels, we relax again the equation to eliminate the components of the error that are represented by high

frequency modes at this level, calculate the residual of the error equation, and project it into the next level to continue eliminating components of the error. When we reach the coarsest grid, we solve exactly the error equation for the error. This operation can be performed at little computational cost since the coarsest level includes a few nodes. The calculated error can be now interpolated to find an additive correction for the solution on the previous level and then a relaxation step is again applied into the solution to ensure the function is smooth enough and can be accurately interpolated. Once the finest level is reached, an additive correction to the solution is found, composed of several terms obtained at each coarse level. At this point, the convergence of the solution is tested and, if the test is not satisfied, a new multilevel iteration is performed.

This algorithm can be summarized as follows:

Relax $\mathbf{A}_0\mathbf{x} = \mathbf{b}$, ν_1 times (ν_1 is usually 1) using the initial value $\mathbf{x}^{(0)}$ as guess.

Compute the residual: $\mathbf{r}_0^{\nu_1} = \mathbf{b} - \mathbf{A}_0\mathbf{x}^{(\nu_1)}$.

Restrict the residual into the next coarse grid: $\mathbf{b}_1 = \mathcal{R}_0^1\mathbf{r}_0$.

Relax $\mathbf{A}_1\mathbf{x}_1 = \mathbf{b}_1$, ν_1 times using $\mathbf{e}_0^{(0)}$ as initial value.

Compute the residual: $\mathbf{r}_1^{\nu_1} = \mathbf{b}_1 - \mathbf{A}_1\mathbf{x}_1^{(\nu_1)}$.

Restrict the residual into the next coarse grid: $\mathbf{b}_2 = \mathcal{R}_1^2\mathbf{r}_1$.

$\vdots$

Solve $\mathbf{A}_n\mathbf{x}_n = \mathbf{b}_n$.

$\vdots$

Interpolate the error and correct the solution: $\mathbf{x}_1^{\nu_1'} = \mathbf{x}_1^{\nu_1} + \mathcal{I}_2^1\mathbf{x}_2$.

Relax $\mathbf{A}_1\mathbf{x}_1 = \mathbf{b}_1$, ν_2 times (ν_2 is usually 1).

Interpolate the error and correct the solution: $\mathbf{x}_0^{\nu_1'} = \mathbf{x}_0^{\nu_1} + \mathcal{I}_1^0\mathbf{x}_1$.

Relax $\mathbf{A}_0\mathbf{x}_0 = \mathbf{b}_0$, ν_2 times

Check the convergence.

The "V" shape of the structure in this algorithm gives the name to this type of multilevel scheme.

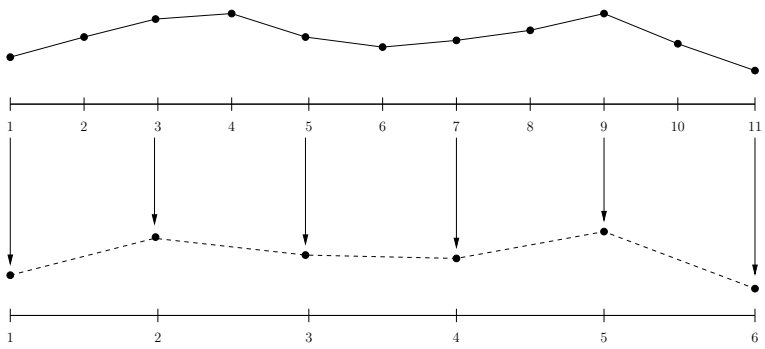


Figure 15: Restriction of an eigenmode with wavenumber $k < n_{coarse}$.

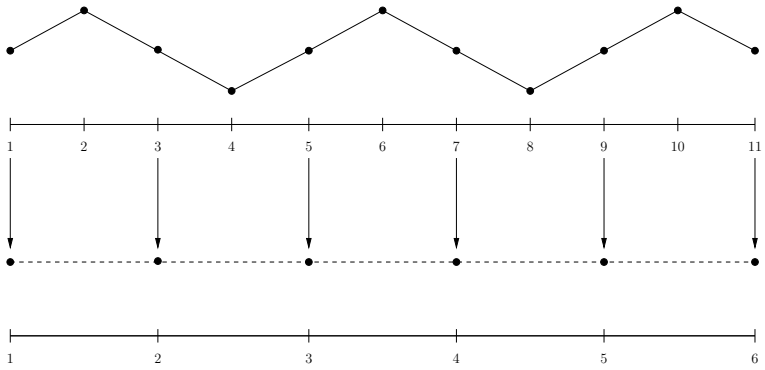


Figure 16: *Aliasing* of an eigenvector with wavenumber $k = n_{coarse}$.

CHAPTER IV

IMPLEMENTATION

4.1 Overview

The implementation of the multilevel solver on EVENT was performed in steps, starting from standalone components and aggregating them into a multilevel transport solver for the in-group equation. This chapter presents some generalities of the implementation of the components first, and then the way these components where integrated into a solver of the in-moment solver which was used first as a diffusion solver and then for the inversion of A_{mm} matrices in a moment by moment block Jacobi scheme.

4.2 Components of multigrid

In order to perform multigrid in EVENT, a set of new subroutines was implemented to perform the tasks necessary to set up and run a V-type multigrid iteration:

a. Meshing and interconnection of different geometrical levels: the preprocessor GEM was modified to allow meshing od multilevel problems.

b. Assembly of coarse grid coefficient matrices: volume weighted homogenization routines were introduced to the preprocessor GEM to generate the material properties for the coarse levels. the coarse grid coefficient matrices are then assembled using the existing EVENT routines.

c. Calculation of the restriction and interpolation operators to communicate the levels: a subroutine to compute the restriction and interpolation operators was implemented in EVENT. This subroutine is called once, at the beginning of a multigrid calculation.

d. Iterative sweep to relax the solution: point Jacobi and Gauss-Seidel iterations were implemented to be used either as a standalone iterative solver, or as part of a multigrid V-cycle.

e. An exact solver to find the correction on the coarser grid: in order to solve the error equation in the coarser grid, a Gauss-Jordan matrix inverting routine was implemented as part of the multigrid solver.

Aside of the meshing and homogenizing routines in GEM and the multigrid solver (MGSOL) in EVENT several minor changes were introduced. Most of them were done to take into account the fact only the first LEVEL_NNOD(1) nodes have an actual physical significance, and then the remaining nodes must not be taken into account for source iterations, eigenvalue calculations, etc.

4.2.1 Multilevel meshing

Multilevel meshing was integrated into the preprocessor, extending the lattice command. Each level is stored as a new geometrical domain, to avoid conflicts with existing solvers. In order to simplify the computation of the restriction / interpolation operators the correlation between levels was done geometrically, using coincidence between fine and coarse meshes (figure 17).

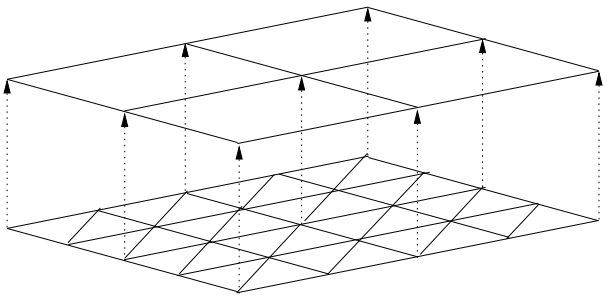


Figure 17: Multilevel meshing as implemented in GEM.

After the geometry is read and the mesh is filled, node numbers are reordered to reduce the bandwidth of the coefficient matrix and minimize the communication between levels. The nodes are separated by level, and the array LEVEL_NNOD(1:NLEVEL) is updated to reflect the maximum node number at each level. The effect, after assembly, is an in-moment coefficient matrix (figure 18) that has in-level diagonal matrix blocks.

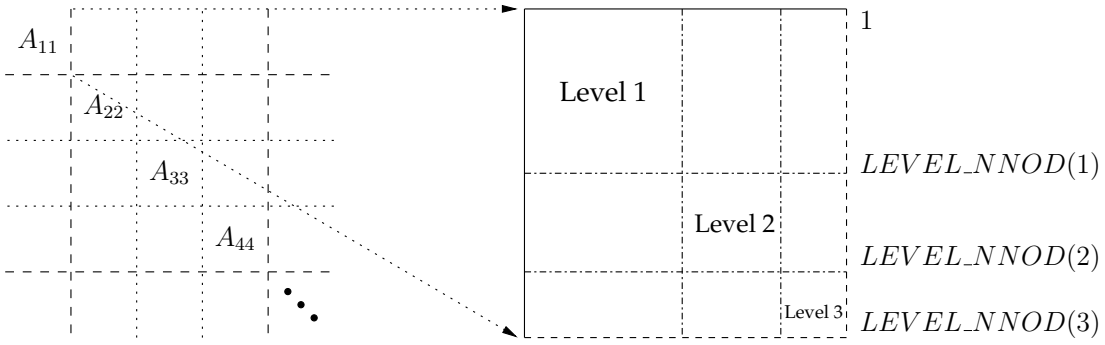


Figure 18: Scheme of the in-moment matrix for multilevel problems.

4.2.2 Homogenization

To produce the homogenized properties needed to assemble the coarse mesh matrices, two homogenization routines were implemented:

- Flux weighted homogenization using the cell command in GEM; and
- Volume weighted homogenization.

The cell command prepares an EVENT input file and runs a cell calculation while preprocessing. Using the produced cell flux distribution, creates a new cell averaged material which then can be assigned to a coarse grid region.

Volume weighted homogenization was implemented element-wise by projecting of the coarse grid element on the fine grid, and then computing the contribution of each fine grid element to the total coarse grid element volume. This produces a new homogenized material per coarse grid element, regardless of the number of material regions defined on the fine grid.

4.2.3 Restriction / interpolation operator

The restriction and interpolation operators are calculated at the same time, as a linear full weighted restriction operator is the transpose of a linearly interpolated prolongation operator. The restriction operator REST_INT is a set of NLEVEL-1 compactly stored $N_c \times N_f$ matrices where each row represents the contribution of each node on the fine grid to a specific point on the coarse grid. The operator is calculated once, the first time the multilevel

solver is called and saved until the end of the execution.

To compute it the node coordinates of the elements that contain coarse grid node N_I are retrieved and a bounding box determined by the maximum and minimum coordinates. Then, the coincident node N_i in the fine grid is searched and also all the nodes connected to it that lay inside of the bounding box. The contribution of each fine node to the coarse node is computed by evaluating the linear base functions of the coarse grid elements at each fine node (figure 19). Finally, each restriction row is normalized to ensure the operator does not modify the norm of the restricted vectors. Although the interpolation operator is the transpose of the restriction operator, the sum of each column that corresponds to a fine node has to be stored separately to normalize the operator during its application.

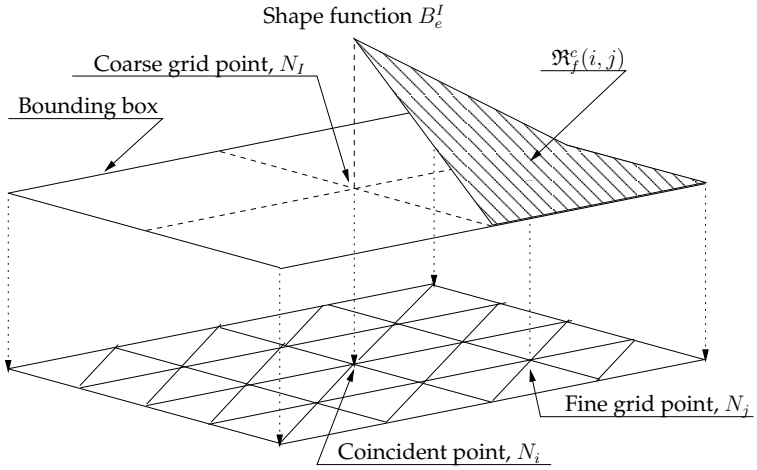


Figure 19: Computation of $\mathfrak{R}_f^c(i,j)$, the contribution of fine node j to coarse node i.

4.2.4 Relaxation iterations

To reduce the cost of updating the solution and compute the residual, the relaxation iterations were reorganized to update the solution and the residual using one matrix-vector multiplication.

First, let us rewrite the expression for a Jacobi iteration (equation (3.7)

$$\begin{aligned} \mathbf{x}^{(k)} = \mathbf{x}^{(k)} &= \mathbf{D}^{-1}\,\mathbf{b} + \mathbf{D}^{-1}\,(\mathbf{L}+\mathbf{U})\,\mathbf{x}^{(k-1)} \\ \mathbf{D}^{-1}\,\mathbf{b} - \mathbf{D}^{-1}\,(\mathbf{D}-\mathbf{L}-\mathbf{U})\,\mathbf{x}^{(k-1)} &+ \mathbf{x}^{(k-1)} \\ \mathbf{x}^{(k)} &= \mathbf{D}^{-1}\,\mathbf{r}^{(k-1)} + \mathbf{x}^{(k-1)} \end{aligned} \tag{4.1}$$

where the residual is updated with:

$$\begin{aligned} \mathbf{r}^{(k)} &= \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)} \\ \mathbf{r}^{(k)} &= \mathbf{b} - \mathbf{A}(\mathbf{D}^{-1}\,\mathbf{r}^{(k-1)} + \mathbf{x}^{(k-1)}) \\ \mathbf{r}^{(k)} &= \mathbf{r}^{(k-1)} - \mathbf{A}\mathbf{D}^{-1}\mathbf{r}^{(k-1)} \end{aligned} \tag{4.2}$$

making the Jacobi iteration with residual update:

$$\begin{aligned} \mathbf{x}^{(k)} &= \mathbf{D}^{-1}\,\mathbf{r}^{(k-1)} + \mathbf{x}^{(k-1)} \\ \mathbf{r}^{(k)} &= \mathbf{r}^{(k-1)} - \mathbf{A}\mathbf{D}^{-1}\mathbf{x}^{(k-1)} \end{aligned} \tag{4.3}$$

In a similar way, Gauss-Seidel iteration can be reformulated as:

$$\begin{aligned} \mathbf{e}^{(k)} &= \mathbf{D}^{-1}(\mathbf{r}^{(k-1)} + \mathbf{L}\mathbf{e}^{(k-1)}) \\ \mathbf{x}^{(k)} &= \mathbf{x}^{(k-1)} + \mathbf{e}^{(k)} \\ \mathbf{r}^{(k)} &= \mathbf{r}^{(k-1)} - (\mathbf{D} - \mathbf{L} - \mathbf{U})\mathbf{e}^{(k)} \end{aligned} \tag{4.4}$$

4.2.5 Coarse grid solver

As coarse grid solver a direct invertion method is used. The first time the solver is called for a given moment, the coarse level coefficient matrix for that moment is inverted and stored. Even being simple, this method is enough for small to medium sized problems, as the number of nodes in the coarse grid is relatively small so the core requirement and computational cost of the inversion is not a burden.

4.3 In-moment (diffusion) solver

A V-cycle multigrid solver was implemented using the components described above. It solves the in-moment problem:

$$\mathbf{A}_{mm}\mathbf{x}_m = \mathbf{b}_m \tag{4.5}$$

which, for $m = 1$ is a diffusion equation solver.

The overall algorithm used is:

1: **if** $rest\_int$ not computed **then**
2: Compute $rest\_int$
3: **end if**

4: $oldresnorm = ||\mathbf{r}_m||_2$

5: $resnorm = ||\mathbf{r}_m||_2$

6: **while** $resnorm > tol \cdot oldresnorm$ **do**

7: **if** coarse grid matrix not inverted **then**

8: Invert and store coarse grid matrix for moment m and group g

9: **end if**

10: **if** $level \neq 1$ **then**

11: $guessflux = 0$

12: **end if**

13: **for** $level = 1$ to $nlevel - 1$ **do**

14: Relax ν_1 times using point Jacobi or Forward Gauss-Seidel iterations

15: Restrict residual to $level + 1$

16: **end for**

17: Solve exactly at the coarsest level

18: **for** $level = nlevel - 1$ down to 1 **do**

19: Correct solution and residual using interpolated solution from $level + 1$

20: Relax ν_2 times using point Jacobi or Backward Gauss-Seidel iterations

21: **end for**

22: **end while**

4.4 Block Jacobi transport solver

To solve the transport equation using the in-moment solver described above, a coupling between the moments is needed. This is performed using the *block Jacobi* method:

1: $oldresnorm = ||\mathbf{r}||_2$

2: **while** $resnorm > tol \cdot oldresnorm$ **do**

3: **for** $m = 1$ to maximum moment **do**

4: $\mathbf{e}_m = \mathbf{A}_{mm}^{-1}\mathbf{r}_m$

5: **end for**

6: $\mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - \mathbf{A}\mathbf{e}^{(k)}$

7: **end while**

The matrix vector multiplication in line 6 is computed using the MATVEC routine.

CHAPTER V

NUMERICAL TESTS

This chapter includes several numerical tests performed to validate the implementation of the solver and to analyze the improvement obtained by rearranging the relaxation iterations into a multigrid solver. In most cases the solution obtained by the new solvers was compared to the solution obtained with using the same code, with a preconditioned conjugate gradient solver. This solver has been benchmarked and validated using both numerical and analytical results under similar conditions [7, 8, 11].

Unless specifically noted, the multigrid solution was obtained using forward Gauss-Seidel iterations as pre-relaxation and backward Gauss-Seidel iterations as post-relaxation.

The solver was tested with the following cases.

- Case 1. Effect of the total cross section:
- Case 2. Effect of the scattering ratio:
- Case 3. Effect of increasing the number of fine meshes per coarse element (2 levels):
- Case 4. Effect of reducing the meshsize adding more levels:
- Case 5. Two group, two region homogenized reactor:
- Case 6. Seven group fuel assembly calculation:
- Case 7: Four region transport problem

Cases 1 to 4 are a parametric analysis of the computational cost, over a simple homogeneous diffusion problem. Cases 5 and 6 are a homogeneous and a heterogeneous diffusion eigenvalue calculation to compare with reference solutions. Finally, Case 7 is a four region P_3 transport problem.

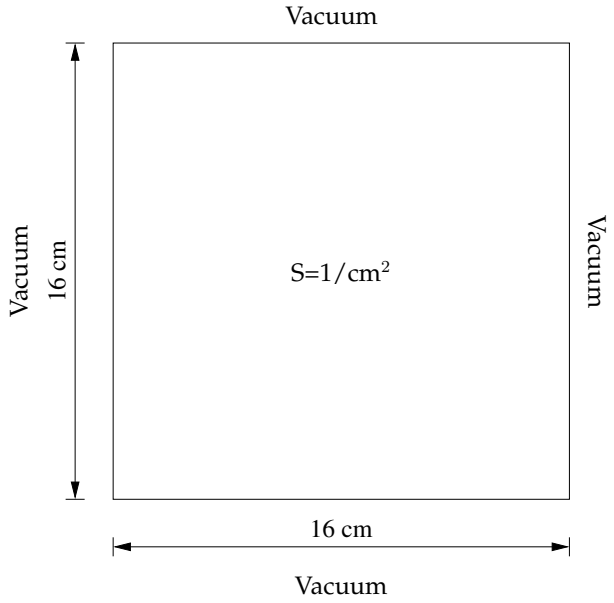


Figure 20: Geometry used in cases 1 to 4.

5.1 In-moment (diffusion) solver

5.1.1 Computational cost analysis

The in-moment solver was tested using a homogeneous problem with vacuum boundary conditions (figure 20). The problem was solved using point Jacobi and Gauss-Seidel iterations, and the multigrid solver. The mesh size, total cross section, and scattering ratio ($c = \sigma_s/\sigma$) were modified to analyze their effect on the computational cost.

The number of relaxation iterations (N_{iter})(or *working units*) and the estimated number of operations were used as measures of computational cost. The number of operations was computed as:

$$N_{ops} \lesssim \frac{N}{1 - 2^{-d}} N_{iter} \tag{5.1}$$

where $d = 2$ is the number of dimensions and N is the number of nodes in the fine grid.

Case 1. Effect of the total cross section: For case 1, a fine mesh with 128×128 nodes and a coarse mesh of 16×16 nodes were used. The scattering ratio was fixed on $c = 0.1$ and the total cross section was given values of 0.10, 0.25, 0.50, 0.75, and 1.00cm^{-1}. The results (table 1) show how the stationary iterations become more and more inefficient as the system becomes more optically thin. The computational cost for multigrid remains

practically constant, with a slight increase as a few more relaxation iterations are necessary to propagate the solution on the fine mesh within each coarse element.

Table 1: Case 1: Varying total cross section (16384 nodes on the fine mesh).

$\sigma[\text{cm}^{-1}]$	**Point Jacobi**		**Gauss-Seidel**		**Multigrid**	
	N_{iter}	N_{ops}	N_{iter}	N_{ops}	N_{iter}	N_{ops}
0.01	44163	7.24E+08	22087	3.62E+08	1642	3.59E+07
0.25	37166	6.09E+08	18587	3.05E+08	1606	3.51E+07
0.50	17745	2.91E+08	8876	1.45E+08	1506	3.29E+07
0.75	10101	1.65E+08	5054	8.28E+07	1370	2.99E+07
1.00	6381	1.05E+08	3194	5.23E+07	1220	2.67E+07

Case 2. Effect of the scattering ratio: Case 2 was run with the same discretization of case 1, a fine mesh with 128×128 nodes and a coarse mesh of 16×16 nodes. The total cross section was fixed at $\sigma = 0.50\text{cm}^{-1}$ and the scattering ratio was assigned values fixed on $c = 0.1$, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 and 1.0.. The results (table 2) show an increase in the number of relaxation iterations needed to converge the problem as the system becomes more diffusive. For low c (absorber medium) there is little coupling between the regions, making the simple iterative solvers good for the job, as little or no information is needed to be transfered across the problem. When c is increased that transfer of information requires a greater number of relaxation iterations but, as the coarse grid remains the same, this does not constitute an important penalty for the multigrid solver.

Case 3. Effect of increasing the number of fine meshes per coarse element (2 levels): Case 3 was set up to compare the effect of a 2 level V-cycle with different grid sizes, preserving the same coarse grid so the cost of the coarse grid solution remains the same. The number of fine elements per coarse element was increased from 16 to 36, 49, 64, 81, and 144. As in case 1, the increasing optical thickness of the problem increases the computational cost slightly when using multigrid, but it still outperforms both Gauss-Seidel and Jacobi iterations.

Table 2: Case 2: Varying the scattering ratio (16384 nodes on the fine mesh).

c	**Point Jacobi**		**Gauss-Seidel**		**Multigrid**	
	N_{iter}	N_{ops}	N_{iter}	N_{ops}	N_{iter}	N_{ops}
0.1	3198	5.2E+07	1603	2.6E+07	936	2.0E+07
0.2	3563	5.8E+07	1785	2.9E+07	982	2.1E+07
0.3	4023	6.6E+07	2015	3.3E+07	1034	2.3E+07
0.4	4618	7.6E+07	2313	3.8E+07	1092	2.4E+07
0.5	5420	8.9E+07	2713	4.4E+07	1152	2.5E+07
0.6	6558	1.1E+08	3283	5.4E+07	1226	2.7E+07
0.7	8303	1.4E+08	4155	6.8E+07	1308	2.9E+07
0.8	11313	1.9E+08	5660	9.3E+07	1400	3.1E+07
0.9	17745	2.9E+08	8876	1.5E+08	1506	3.3E+07
1.0	41128	6.7E+08	20568	3.4E+08	1628	3.6E+07

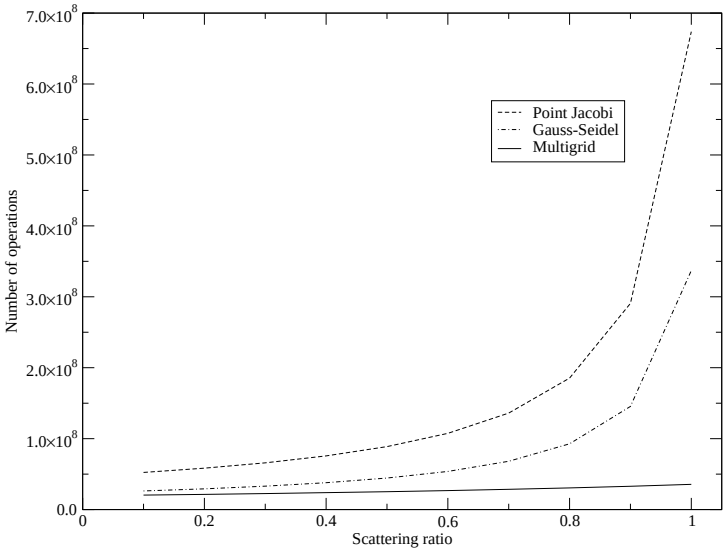


Figure 21: Case 2: Varying the scattering ratio (16384 nodes on the fine mesh).

Table 3: Case 3: Increasing the number of fine meshes per coarse element (2 levels).

$\frac{h_{coarse}}{hfine}$	N	**Point Jacobi**		**Gauss-Seidel**		**Multigrid**	
		N_{iter}	N_{ops}	N_{iter}	N_{ops}	N_{iter}	N_{ops}
4	4096	11043	4.5E+07	5325	2.2E+07	406	2217301
6	9216	24843	2.3E+08	12426	1.1E+08	924	1.1E+07
7	12544	33813	4.2E+08	16911	2.1E+08	1258	2.1E+07
8	16384	44163	7.2E+08	22087	3.6E+08	1642	3.6E+07
9	20736	55893	1.2E+09	27952	5.8E+08	2074	5.7E+07
10	25600	69003	1.8E+09	34507	8.8E+08	2558	8.7E+07

Case 4. Effect of reducing the meshsize adding more levels: Case 4 was run to test to which order of the node count the computational cost increases. As shown above, the cost of a multigrid iteration depends on the optical thickness of each coarse element, so the cost of the solution on the coarsest level was fixed by maintaining the same 8×8 mesh. The grid was refined by a binary segmentation on each coordinate:

- 256 nodes: 16×16 nodes in 1 level; 16×16, 8×8 nodes in 2 levels.
- 1024 nodes: 32×32 nodes in 1 level; 32×32, 16×16, 8×8 nodes in 3 levels.
- 4096 nodes: 64×64 nodes in 1 level; 64×64, 32×32, 16×16, 8×8 nodes in 4 levels.
- 16384 nodes: 128×128 nodes in 1 level; 128×128, 64×64, 32×32, 16×16, 8×8 nodes in 5 levels.

The results show (Table 4, Figure 22), for the same fixed tolerance of 10^{-6}, Gauss-Seidel and Jacobi methods require $O(N^{2.00})$ operations, whereas multigrid with one pre and post relaxation requires $O(N^{1.76})$. If we now increase the number of pre and post-relaxation iterations per level to 8, the order of the computational cost is reduced to 1.58

Table 4: Case 4: Reducing the meshsize adding more levels.

N	**Point Jacobi**		**Gauss-Seidel**		**Multigrid** $\nu_1 = \nu_2 = 1$		**Multigrid** $\nu_1 = \nu_2 = 8$	
	N_{iter}	N_{ops}	N_{iter}	N_{ops}	N_{iter}	N_{ops}	N_{iter}	N_{ops}
256	415	1.06E+05	826	2.11E+05	114	3.89E+04	352	1.20E+05
1024	1665	1.70E+06	3305	3.38E+06	314	4.29E+05	640	8.74E+05
4096	6610	2.71E+07	13214	5.41E+07	896	4.89E+06	1536	8.39E+06
16384	26429	4.33E+08	52848	8.66E+08	2684	5.86E+07	3968	8.67E+07

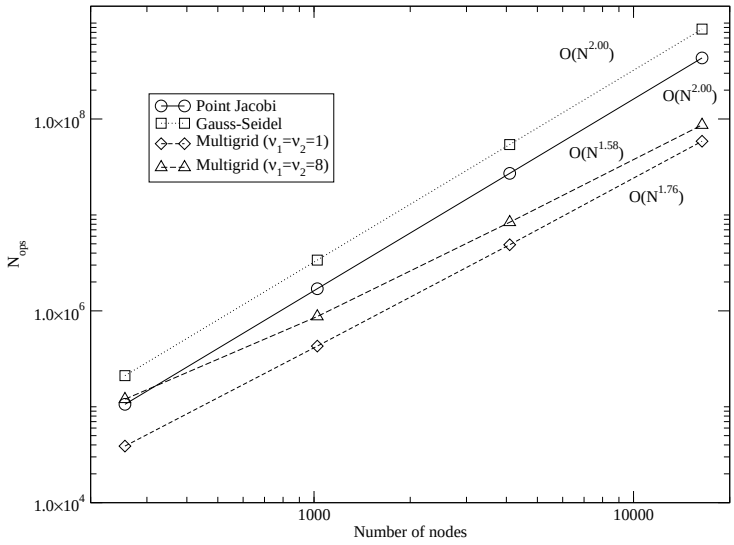


Figure 22: Case 4: Reducing the meshsize adding more levels.

5.1.2 Eigenvalue calculations

Case 5. Two group, three region homogenized reactor: To test the solver within an eigenvalue iteration, a simple two group, three region diffusion problem [14] was used. The geometry of the problem is shown on Figure 23 and the cross sections in Table 5. The geometry was discretized in $78 \times 144 = 11232$ nodes, using a 13×24 and a 6×12 meshes as coarse grids.

The multigrid, Gauss-Seidel, and point Jacobi solvers converged to the reference solution obtained with the conjugate gradient solver. The multiplication factor for the diffusion solution (Figure 24) was $k_{eff} = 1.14223$, and was obtained using 60128, 93536, and 186222 relaxation iterations, respectively.

Case 6. Seven group fuel assembly calculation: Case 6 is a 17×17 UO_2 fuel pin PWR assembly (Figure 25), taken from the C5G7 MOX benchmark specification [8, 10]. The geometry was discretized using $h \simeq 0.25$cm (5 elements per cell side, 4 elements per pin quadrant), resulting in a 10864 nodes fine mesh. A reference solution was obtained using the CGSOL5 preconditioned conjugate gradient solver, resulting in a multiplication factor $k_{eff} = 1.33737$.

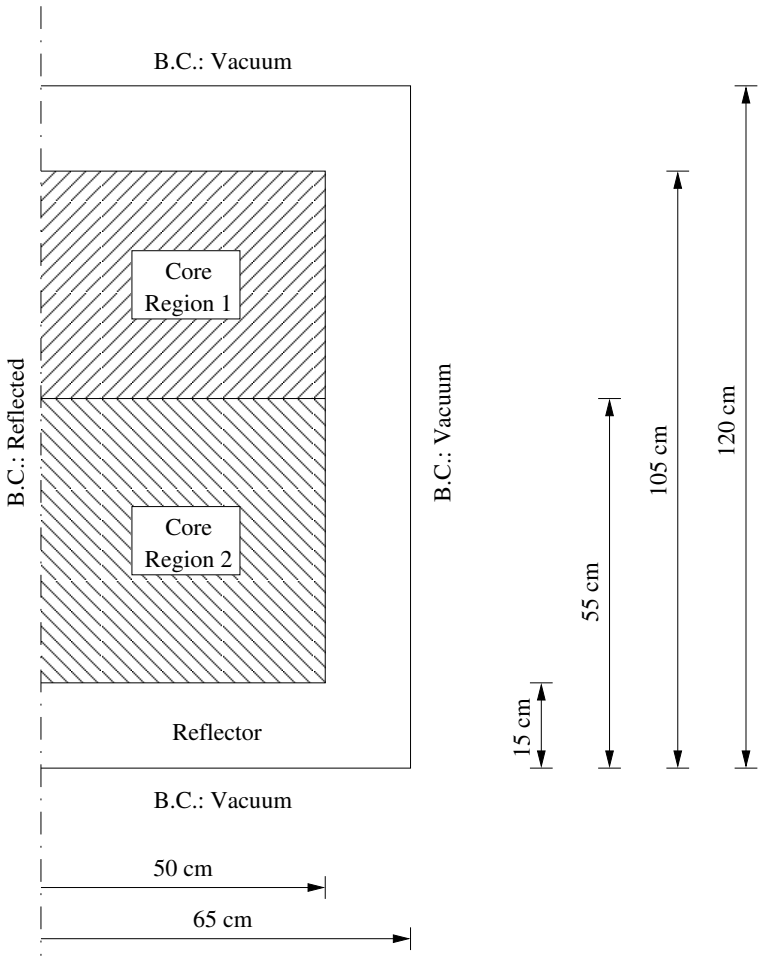
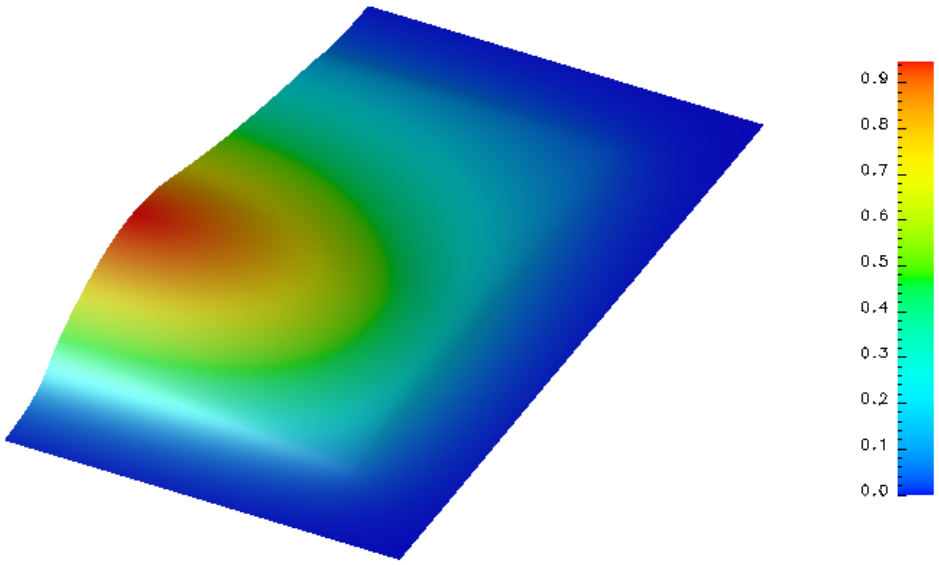


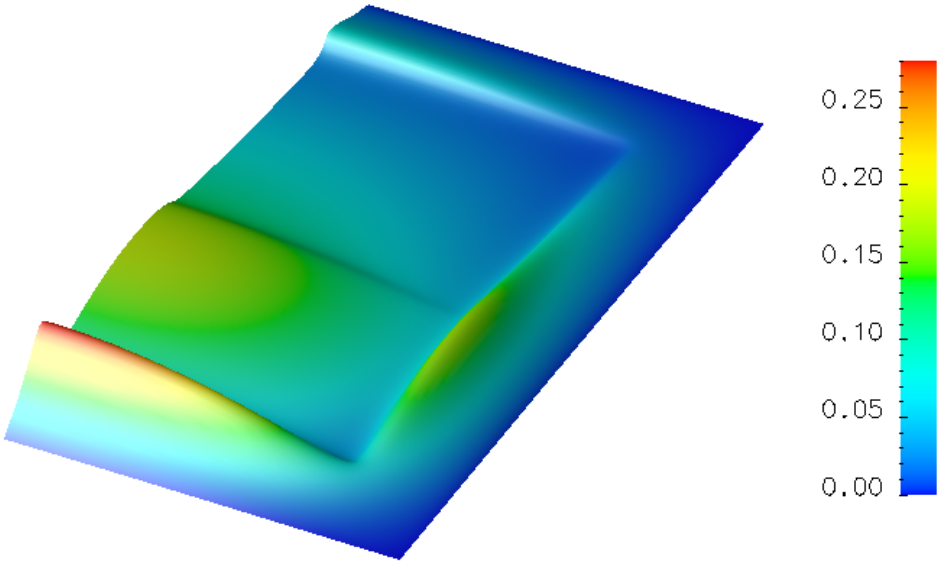
Figure 23: Geometry for case 5.

Table 5: Material data for case 5.

		Core region 1		**Core region 2**		**Reflector**	
		Group 1	Group 2	Group 1	Group 2	Group 1	Group 2
σ_t^g		0.2631	0.9416	0.2604	0.8333	0.2950	2.0080
σ_a^g		0.0121	0.1210	0.0100	0.1000	0.0004	0.0200
$\nu\sigma_f^g$		0.0085	0.1851	0.0060	0.1500	0.0000	0.0000
χ_g		1.0000	0.0000	1.0000	0.0000	0.0000	0.0000
$\sigma_s^{g\to g'}$	Group 1	0.2269	0.0000	0.2344	0.0000	0.2453	0.0000
	Group 2	0.0241	0.8206	0.0160	0.7333	0.0493	1.9880



Group 1



Group 2

Figure 24: Scalar flux for case 5.

The unaccelerated solvers needed 417992 (Jacobi) and 210832 (Gauss-Seidel) relaxation iterations to converge to the desired tolerance. The multigrid solver was run with a three level structure with 18×18 nodes and 9×9 nodes in the coarse levels, converging in 94528 relaxation iterations.

The scalar flux maps for each group (Figure 26) show the shift in maximum flux from the center of the fuel pins to the moderator that surrounds the pins and fills the guide tubes.

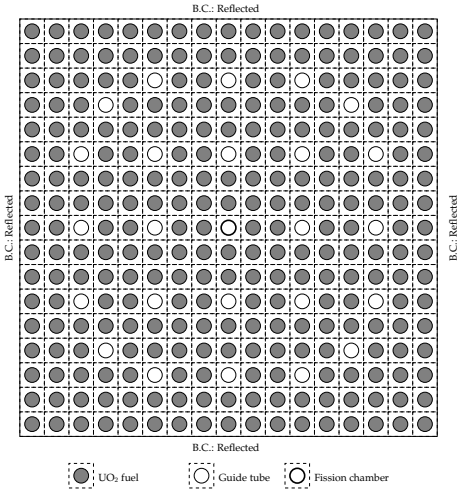
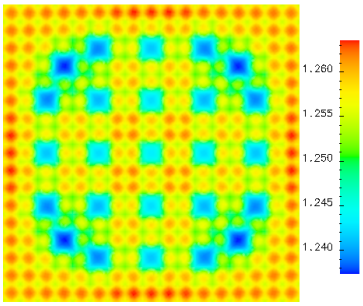
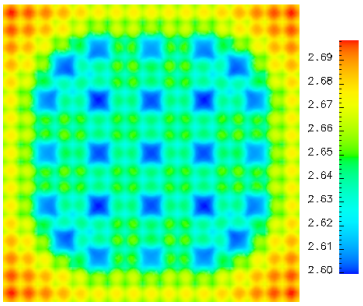


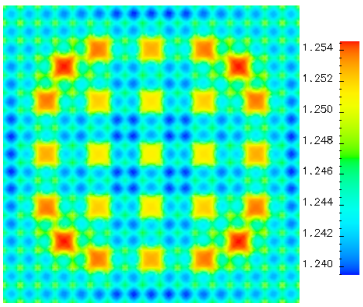
Figure 25: Material distribution for the UO_2 PWR fuel assembly used in case 6.



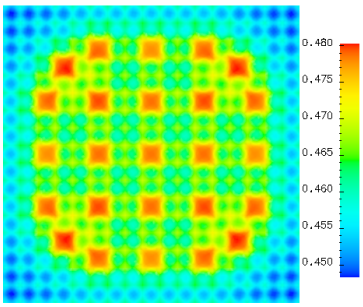
Group 1



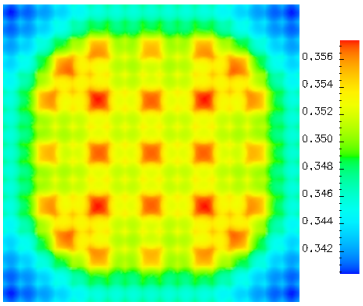
Group 2



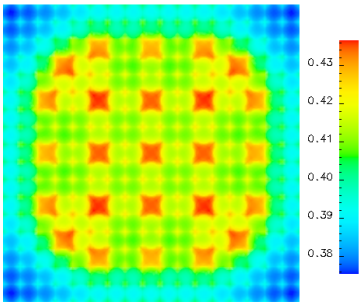
Group 3



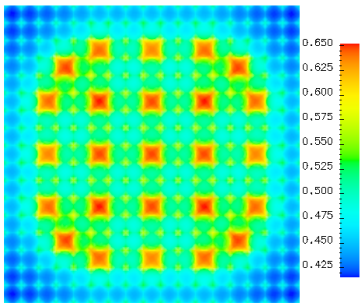
Group 4



Group 5



Group 6



Group 7

Note: Scale changes from group to group

Figure 26: Scalar flux map for case 6.

5.2 *Transport solver*

Case 7: Four region transport problem: The transport solver was tested with a four region problem (Figure 27). One of the quarters has a strong absorbing material, whereas in the other three the scattering is prevalent. Opposing to the absorbing quarter there is a source, which creates a gradient in the neutron flux. The system is surrounded by vacuum boundary conditions.

As in the previous cases, a reference solution was obtained using the CGSOL5 solver with local tolerance 4 orders of magnitude smaller. The moments of the even parity flux obtained with the multigrid solver (Figure 28) were compared with the reference solution (Figure 29). From these plots we can see the all the moments of the solution are smooth, i.e. no artifacts are introduced by the correction in the coarse level.

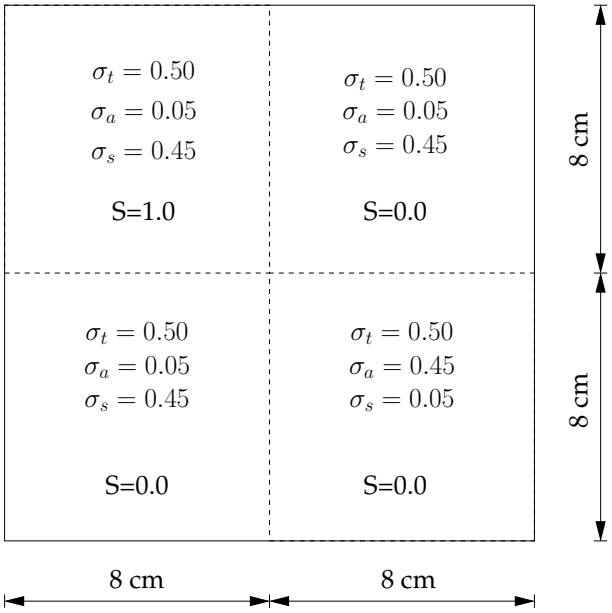


Figure 27: Geometry and material distribution for case 7.

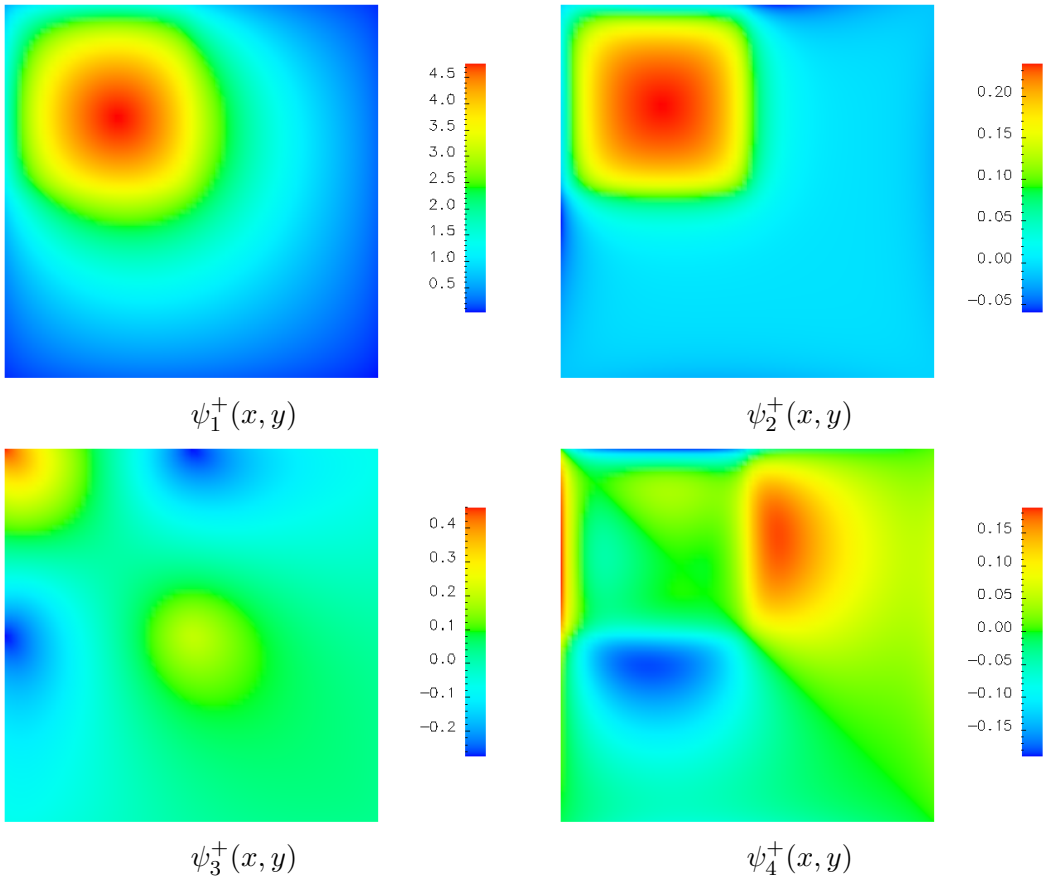


Figure 28: Moments of the even parity flux for case 7.

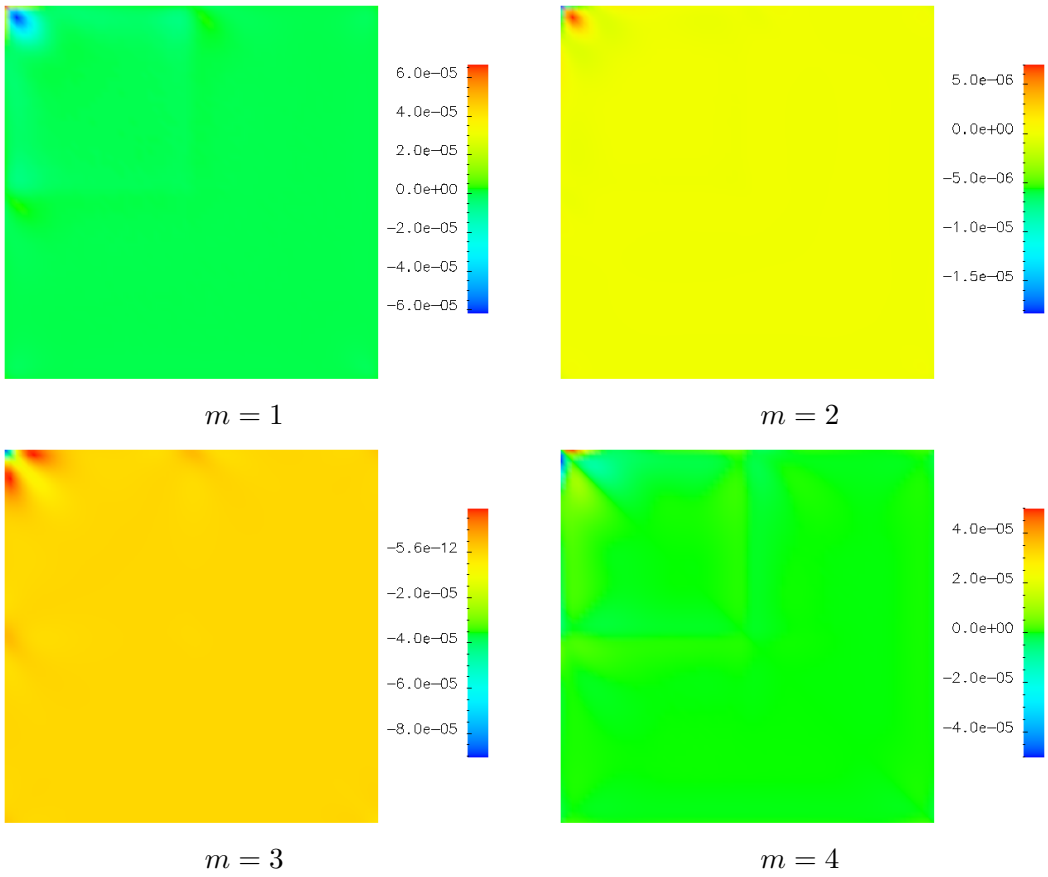


Figure 29: Difference between the solution obtained with the multigrid solver, and the preconditioned conjugate gradient solution for case 7.

CHAPTER VI

CONCLUSIONS

A proof of concept multigrid solver for the even parity second order transport equation was implemented using the infrastructure provided by the finite element, spherical harmonics code EVENT. The solver was implemented for the solution diffusion-like inmoment equation and a block Jacobi scheme was used to update the residual in higher order problems. The solver was tested satisfactorily for the diffusion and transport equation with fixed source and eigenvalue problems.

The results show the implemented multilevel scheme outperforms the direct application of the relaxation iterations, both in total computational cost and in the order on which the cost scales with the number of nodes. This difference is more important for optically thin, highly scattering (diffusive) problems where the one-level solver convergence is particularly low.

The rather simple relaxation iteration and coarse grid solver used limit the overall efficiency of the method, which still cannot be compared with the highly optimized preconditioned conjugate gradient solver already implemented implemented in the code. Nevertheless, the positive results obtained with these simple tools makes interesting the research on the combination of a hybrid method, either using multigrid as a preconditioner for the conjugate gradient solver or using the conjugate gradient method for the coarse grid solver.

REFERENCES

[1] ACKROYD, R.T., *Finite Element Methods for Particle Transport: Applications to Reactor and Radiation Physics*. Wiley, 1996.

[2] ADAMS, M.L. AND LARSEN, E.W., "Fast iterative methods for discrete-ordinates particle transport calculations," *Progress in Nuclear Energy*, vol. 40, no. 1, pp. 3–159, 2002.

[3] ALCOUFFE, R.E., "A Stable Diffusion Synthetic Acceleration Method for Neutron Transport Iterations," *Trans. Am. Nucl. Soc*, vol. 23, p. 203, 1976.

[4] ALCOUFFE, R.E., "Diffusion synthetic acceleration methods for the diamond differenced discrete-ordinates problem," *Nucl. Sci. Eng*, vol. 64, pp. 344–355, 1977.

[5] BARNETT, A, MOREL J.E. AND HARRIS D.R., "A Multigrid Acceleration Method for the 1-D SN Equations with Anisotropic Scattering," *Nucl. Sci. Eng*, vol. 102, p. 1, 1989.

[6] DE OLIVEIRA, C.R.E., "An arbitrary geometry finite element method for multigroup neutron transport with anisotropic scattering," *Prog. Nucl. Energy*, vol. 18, no. 1/2, pp. 227–236, 1986.

[7] DE OLIVEIRA, C.R.E., *Finite element techniques for multi-group neutron transport calculations with anisotropic scattering*. PhD thesis, Queen Mary College, University of London, UK, 1987.

[8] KELLER, S.E. AND DE OLIVEIRA, C.R.E., "Two-Dimensional C5G7 Mox Fuel Assembly Benchmark Calculations Using the FEM-PN Code EVENT," *Prog. In Nuclear Energy*, vol. 45, p. 255, 2004.

[9] LARSEN, E.W., "Transport acceleration methods as two-level multigrid algorithms," *Modern mathematical methods in transport theory, Proc. 11th Int. Conf. Symp., Blacksburg/VA (USA)*, 1989.

[10] LEWIS, EE AND SMITH, MA AND PALMIOTTI, G. AND TAIWO, TA AND TSOULFANIDIS, N., "Benchmark specification for Deterministic 2-D/3-D MOX fuel assembly transport calculations without spatial homogenisation (C5G7 MOX)," tech. rep., NEA/NSC/DOC, 4, 2001.

[11] PARK, H.K., *Coupled Space-Angle Adaptivity and Goal-Oriented Error Control for Radiation Transport Calculations*. PhD thesis, Georgia Institute of Technology, Atlanta, GA, 2006.

[12] RAMONE, G.L. AND ADAMS, M.L. AND NOWAK, P.F., "A Transport Synthetic Acceleration Method for Transport Iterations," *Nucl. Sci. Eng*, vol. 125, pp. 257–283, 1997.

[13] SAAD, Y., *Iterative methods for sparse linear systems*. SIAM, 2003.

[14] STACEY, W.M., *Nuclear Reactor Physics*. Wiley-Interscience, 2001.

[15] TROTTENBERG, U. AND OOSTERLEE, C.W., *Multigrid*. Academic Press, 2001.

[16] WOOD, J. AND WILLIAMS, M.M.R., "Recent Progress in the Application of the Finite Element Method to the Neutron Transport Equation," *Prog. Nucl. Energy*, vol. 4, pp. 21–40, 1984.